Package 'tablespan'

September 10, 2025

Type Package

Title Create Satisficing 'Excel', 'HTML', 'LaTeX', and 'RTF' Tables using a Simple Formula

Version 0.3.2

Maintainer Jannik H. Orzek <jannik.orzek@mailbox.org>

Description Create ``good enough'' tables with a single formula. 'tablespan' tables can be exported to 'Excel', 'HTML', 'LaTeX', and 'RTF' by leveraging the packages 'openxlsx' and 'gt'. See <https://jhorzek.github.io/tablespan/> for an introduction.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.2

Suggests testthat (>= 3.0.0)

Depends R (>= 4.2.0)

Config/testthat/edition 3

Imports dplyr, gt, methods, openxlsx, rlang, scales, tibble, utils

URL https://github.com/jhorzek/tablespan, https://jhorzek.github.io/tablespan/

BugReports https://github.com/jhorzek/tablespan/issues

NeedsCompilation no

Author Jannik H. Orzek [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-3123-2248>)

Repository CRAN

Date/Publication 2025-09-10 07:50:02 UTC

Contents

`as_excel` *as_excel*

Description

Write a tablespan table to an excel workbook.

Usage

```
as_excel(
  tbl,
  workbook = openxlsx::createWorkbook(),
  sheet = "Table",
  start_row = 1,
  start_col = 1,
  merge_rownames = TRUE
)
```

Arguments

`tbl`	table created with tablespan::tablespan
`workbook`	Excel workbook created with openxlsx::createWorkbook()
`sheet`	name of the sheet to which the table should be written to
`start_row`	row at which to start the table
`start_col`	column at which to start the table
`merge_rownames`	should row names with identical entries be merged?

Value

openxlsx workbook object that can be edited and saved with openxlsx

Examples

```
library(tablespan)
library(dplyr)
data("iris")

tbl <- tablespan(data = iris[iris$Species == "setosa", ],
          formula = Species ~ (Sepal = Sepal.Length + Sepal.Width) +
            (Petal = (Width = Petal.Length) + Petal.Width))

wb <- as_excel(tbl = tbl)

# saveWorkbook(wb, "iris.xlsx")

# The main use case for tablespan is when you already have a summarized table
# that you now want to share using xlsx. The following shows an example using
# the dplyr package:

# First summarize the data:
summarized_table <- mtcars |>
  group_by(cyl, vs) |>
  summarise(N = n(),
            mean_hp = mean(hp),
            sd_hp = sd(hp),
            mean_wt = mean(wt),
            sd_wt = sd(wt))

# Now, we want to create a table, where we show the grouping variables
# as row names and also create spanners for the horse power (hp) and the
# weight (wt) variables:
tbl <- tablespan(data = summarized_table,
          formula = Cylinder:cyl + Engine:vs ~
            N +
            (`Horse Power` = Mean:mean_hp + SD:sd_hp) +
            (`Weight` = Mean:mean_wt + SD:sd_wt),
          title = "Motor Trend Car Road Tests",
          subtitle = "A table created with tablespan",
          footnote = "Data from the infamous mtcars data set.")

wb <- as_excel(tbl = tbl)

# Create the excel table:
# openxlsx::saveWorkbook(wb,
#                        file = "cars.xlsx", overwrite = TRUE)
```

---

as_gt *as_gt*

Description

Translates a table created with tablespan to a great table (gt). See <https://gt.rstudio.com/>.

Usage

```
as_gt(
  tbl,
  groupname_col = NULL,
  separator_style = gt::cell_borders(sides = c("right"), weight = gt::px(1), color =
    "gray"),
  auto_format = TRUE,
  ...
)
```

Arguments

tbl	table created with tablespan::tablespan
groupname_col	Provide column names to group data. See ?gt::gt for more details.
separator_style	style of the vertical line that separates the row names from the data.
auto_format	should the table be formatted automatically?
...	additional arguments passed to gt::gt().

Details

Tablespan itself does not provide any printing of tables as HTML table. However, with as_gt, tablespan can be translated to a great table which provides html and LaTeX output.

Value

gt table that can be further adapted with the gt package.

Examples

```
library(tablespan)
library(dplyr)
data("mtcars")

summarized_table <- mtcars |>
  group_by(cyl, vs) |>
  summarise(N = n(),
            mean_hp = mean(hp),
            sd_hp = sd(hp),
            mean_wt = mean(wt),
            sd_wt = sd(wt))

tbl <- tablespan(data = summarized_table,
                 formula = (LHS = Cylinder:cyl + Engine:vs) ~
                   N +
                   (Results = (`Horse Power` = Mean:mean_hp + SD:sd_hp) +
                      (`Weight` = Mean:mean_wt + SD:sd_wt)))

gt_tbl <- as_gt(tbl)
gt_tbl
```

format_column *format_column*

Description

Change the formatting of a column or single cells within columns.

Usage

```
format_column(
  tbl,
  columns = dplyr::everything(),
  rows = NULL,
  format_gt = gt::fmt_auto,
  format_openxlsx = "GENERAL",
  stack = TRUE
)
```

Arguments

tbl	tablespan table
columns	the columns to style. Must be a tidyselect selector expression (e.g., starts_with("hp_"))
rows	indices of the rows which should be styled. When set to NULL, the style is applied to all rows
format_gt	formatting used for gt. This must be a function with the following signature: function(tbl, columns, rows, ...) and return the tbl with applied formatting. See examples.
format_openxlsx	an argument passed to the numFmt field for openxlsx::createStyle.
stack	When set to TRUE, the style is added on top of the existing styles. This is mostly relevant for openxlsx. When set to FALSE, the new style replaces all previous styling.

Value

the tablespan table with added styles

Examples

```
library(tablespan)
library(dplyr)
data("mtcars")

# We want to report the following table:
summarized_table <- mtcars |>
  group_by(cyl, vs) |>
```

```
  summarise(N = n(),
            mean_hp = mean(hp),
            sd_hp = sd(hp),
            mean_wt = mean(wt),
            sd_wt = sd(wt))

# Create a tablespan:
tbl <- tablespan(data = summarized_table,
                 formula = Cylinder:cyl + Engine:vs ~
                   N +
                   (`Horse Power` = Mean:mean_hp + SD:sd_hp) +
                   (`Weight` = Mean:mean_wt + SD:sd_wt),
                 title = "Motor Trend Car Road Tests",
                 subtitle = "A table created with tablespan",
                 footnote = "Data from the infamous mtcars data set.")

tbl |>
  format_column(columns = mean_hp,
                rows = c(1,3),
                format_gt = function(tbl, columns, rows, ...){
                             return(gt::fmt_number(tbl,
                                       columns = columns,
                                       rows = rows,
                                       decimals = 4))},
                format_openxlsx = "0.0000") |>
  as_gt()
```

`print.Tablespan` *print.Tablespan*

Description

print.Tablespan

Usage

```
## S3 method for class 'Tablespan'
print(x, digits = 2, n = 3, ...)
```

Arguments

`x`	result from tablespan
`digits`	number of digits to round doubles to
`n`	number of rows to print
`...`	additional arguments passed to prmatrix

Value

nothing

Examples

```
data("iris")
tbl <- tablespan(data = iris[iris$Species == "setosa", ],
          formula = Species ~ (Sepal = Sepal.Length + Sepal.Width) +
            (Petal = Petal.Length + Petal.Width))
print(tbl)
```

`style_column` *style_column*

Description

Change the style of a column or single cells within columns.

Usage

```
style_column(
  tbl,
  columns = dplyr::everything(),
  rows = NULL,
  background_color = "#ffffff",
  text_color = "#000000",
  font_size = NULL,
  bold = FALSE,
  italic = FALSE,
  color_scale = NULL,
  openxlsx_style = NULL,
  gt_style = NULL,
  stack = TRUE
)
```

Arguments

`tbl`	tablespan table
`columns`	the columns to style. Must be a tidyselect selector expression (e.g., starts_with("hp_"))
`rows`	indices of the rows which should be styled. When set to NULL, the style is applied to all rows
`background_color`	hex code for the background color
`text_color`	hex code for the text color
`font_size`	font size
`bold`	set to TRUE for bold
`italic`	set to TRUE for italic

color_scale	a named vector of length 2 or 3 to define a color scale. Example for two colors: color_scale = c("#EE2F43" = -1, "#37E65A" = 1). Example for three colors: color_scale = c("#EE2F43" = -1, "#FFFFFF" = 0, "#37E65A" = 1)
openxlsx_style	optional custom openxlsx style. When provided, all other arguments are ignored
gt_style	optional custom gt style. When provided, all other arguments are ignored
stack	When set to TRUE, the style is added on top of the existing styles. This is mostly relevant for openxlsx. When set to FALSE, the new style replaces all previous styling.

Value

the tablespan table with added styles

Examples

```
library(tablespan)
library(dplyr)
data("mtcars")

# We want to report the following table:
summarized_table <- mtcars |>
  group_by(cyl, vs) |>
  summarise(N = n(),
            mean_hp = mean(hp),
            sd_hp = sd(hp),
            mean_wt = mean(wt),
            sd_wt = sd(wt))

# Create a tablespan:
tbl <- tablespan(data = summarized_table,
                 formula = Cylinder:cyl + Engine:vs ~
                   N +
                   (`Horse Power` = Mean:mean_hp + SD:sd_hp) +
                   (`Weight` = Mean:mean_wt + SD:sd_wt),
                 title = "Motor Trend Car Road Tests",
                 subtitle = "A table created with tablespan",
                 footnote = "Data from the infamous mtcars data set.")

tbl |>
  style_column(columns = mean_hp,
               bold = TRUE) |>
  as_gt()
```

---

style_footnote *style_footnote*

Description

Set the style used for the footnote of the tablespan table.

Usage

```
style_footnote(
  tbl,
  background_color = "#ffffff",
  text_color = "#000000",
  font_size = NULL,
  bold = FALSE,
  italic = FALSE,
  openxlsx_style = NULL,
  gt_style = NULL
)
```

Arguments

tbl	tablespan table
background_color	hex code for the background color
text_color	hex code for the text color
font_size	font size
bold	set to TRUE for bold
italic	set to TRUE for italic
openxlsx_style	optional custom openxlsx style. When provided, all other arguments are ignored
gt_style	optional custom gt style. When provided, all other arguments are ignored

Details

The styling for openxlsx and gt works differently:

- openxlsx_style must be a style object created with openxlsx::createStyle. This style will then be applied to the footnote - gt_style must be a list of gt::tab_style objects to be applied to the table

Value

the tablespan table with added styles

Examples

```
library(tablespan)
library(dplyr)
data("mtcars")

# We want to report the following table:
summarized_table <- mtcars |>
  group_by(cyl, vs) |>
  summarise(N = n(),
            mean_hp = mean(hp),
            sd_hp = sd(hp),
            mean_wt = mean(wt),
```

```
            sd_wt = sd(wt))

# Create a tablespan:
tbl <- tablespan(data = summarized_table,
                 formula = Cylinder:cyl + Engine:vs ~
                   N +
                   (`Horse Power` = Mean:mean_hp + SD:sd_hp) +
                   (`Weight` = Mean:mean_wt + SD:sd_wt),
                 title = "Motor Trend Car Road Tests",
                 subtitle = "A table created with tablespan",
                 footnote = "Data from the infamous mtcars data set.")

tbl |>
  style_footnote(bold = TRUE) |>
  as_gt()
```

`style_header` *style_header*

Description

Set the style used for the header of the tablespan table.

Usage

```
style_header(
  tbl,
  background_color = "#ffffff",
  text_color = "#000000",
  font_size = NULL,
  bold = FALSE,
  italic = FALSE,
  openxlsx_style = NULL,
  gt_style = NULL
)
```

Arguments

`tbl`	tablespan table
`background_color`	hex code for the background color
`text_color`	hex code for the text color
`font_size`	font size
`bold`	set to TRUE for bold
`italic`	set to TRUE for italic
`openxlsx_style`	optional custom openxlsx style. When provided, all other arguments are ignored
`gt_style`	optional custom gt style. When provided, all other arguments are ignored

Details

The styling for openxlsx and gt works differently:

- openxlsx_style must be a style object created with openxlsx::createStyle. This style will then be applied to the header - gt_style must be a list of gt::tab_style objects to be applied to the table

Value

the tablespan table with added styles

Examples

```
library(tablespan)
library(dplyr)
data("mtcars")

# We want to report the following table:
summarized_table <- mtcars |>
  group_by(cyl, vs) |>
  summarise(N = n(),
            mean_hp = mean(hp),
            sd_hp = sd(hp),
            mean_wt = mean(wt),
            sd_wt = sd(wt))

# Create a tablespan:
tbl <- tablespan(data = summarized_table,
                 formula = Cylinder:cyl + Engine:vs ~
                   N +
                   (`Horse Power` = Mean:mean_hp + SD:sd_hp) +
                   (`Weight` = Mean:mean_wt + SD:sd_wt),
                 title = "Motor Trend Car Road Tests",
                 subtitle = "A table created with tablespan",
                 footnote = "Data from the infamous mtcars data set.")

tbl |>
  style_header(
    openxlsx_style = openxlsx::createStyle(
      fontSize = 8,
      fgFill = "#ffffff"),
    gt_style = list(gt::cell_text(size = 8))) |>
  as_gt()
```

`style_header_cells` *style_header_cells*

Description

Set the style used for the cells in the openxlsx export. This function is used to create the borders around cells in openxlsx.

Usage

```
style_header_cells(
  tbl,
  background_color = "#ffffff",
  text_color = "#000000",
  font_size = NULL,
  bold = FALSE,
  italic = FALSE,
  openxlsx_style = NULL
)
```

Arguments

tbl	tablespan table
background_color	hex code for the background color
text_color	hex code for the text color
font_size	font size
bold	set to TRUE for bold
italic	set to TRUE for italic
openxlsx_style	optional custom openxlsx style. When provided, all other arguments are ignored

Details

- openxlsx_style must be a style object created with openxlsx::createStyle. This style will then be applied to the header

Value

the tablespan table with added styles

Examples

```
library(tablespan)
library(dplyr)
data("mtcars")

# We want to report the following table:
summarized_table <- mtcars |>
  group_by(cyl, vs) |>
  summarise(N = n(),
            mean_hp = mean(hp),
            sd_hp = sd(hp),
            mean_wt = mean(wt),
            sd_wt = sd(wt))

# Create a tablespan:
tbl <- tablespan(data = summarized_table,
```

```
                  formula = Cylinder:cyl + Engine:vs ~
                    N +
                    (`Horse Power` = Mean:mean_hp + SD:sd_hp) +
                    (`Weight` = Mean:mean_wt + SD:sd_wt),
                  title = "Motor Trend Car Road Tests",
                  subtitle = "A table created with tablespan",
                  footnote = "Data from the infamous mtcars data set.")

wb <- tbl |>
  style_header_cells(text_color = "#345364") |>
  as_excel()
# save workbook to see the effect
```

`style_hline` *style_hline*

Description

Set the style used for the horizontal lines of the tablespan table. Currently only supported for excel export.

Usage

```
style_hline(tbl, openxlsx_style)
```

Arguments

`tbl`	tablespan table
`openxlsx_style`	style used when exporting to openxlsx

Details

- openxlsx_style must be a style object created with openxlsx::createStyle. This style will then be applied to the horizontal lines

Value

the tablespan table with added styles

Examples

```
library(tablespan)
library(dplyr)
data("mtcars")

# We want to report the following table:
summarized_table <- mtcars |>
  group_by(cyl, vs) |>
  summarise(N = n(),
```

```
            mean_hp = mean(hp),
            sd_hp = sd(hp),
            mean_wt = mean(wt),
            sd_wt = sd(wt))

# Create a tablespan:
tbl <- tablespan(data = summarized_table,
                 formula = Cylinder:cyl + Engine:vs ~
                   N +
                   (`Horse Power` = Mean:mean_hp + SD:sd_hp) +
                   (`Weight` = Mean:mean_wt + SD:sd_wt),
                 title = "Motor Trend Car Road Tests",
                 subtitle = "A table created with tablespan",
                 footnote = "Data from the infamous mtcars data set.")

wb <- tbl |>
  style_hline(
    openxlsx_style = openxlsx::createStyle(
      border = "Top",
      borderColour = "#928505",
      borderStyle = "thin")) |>
  as_excel()
# save workbook to see effect
```

`style_subtitle` *style_subtitle*

Description

Set the style used for the subtitle of the tablespan table.

Usage

```
style_subtitle(
  tbl,
  background_color = "#ffffff",
  text_color = "#000000",
  font_size = NULL,
  bold = FALSE,
  italic = FALSE,
  openxlsx_style = NULL,
  gt_style = NULL
)
```

Arguments

`tbl`	tablespan table
`background_color`	hex code for the background color

text_color	hex code for the text color
font_size	font size
bold	set to TRUE for bold
italic	set to TRUE for italic
openxlsx_style	optional custom openxlsx style. When provided, all other arguments are ignored
gt_style	optional custom gt style. When provided, all other arguments are ignored

Details

The styling for openxlsx and gt works differently:

- openxlsx_style must be a style object created with openxlsx::createStyle. This style will then be applied to the subtitle - gt_style must be a list of gt::tab_style objects to be applied to the table

Value

the tablespan table with added styles

Examples

```
library(tablespan)
library(dplyr)
data("mtcars")

# We want to report the following table:
summarized_table <- mtcars |>
  group_by(cyl, vs) |>
  summarise(N = n(),
            mean_hp = mean(hp),
            sd_hp = sd(hp),
            mean_wt = mean(wt),
            sd_wt = sd(wt))

# Create a tablespan:
tbl <- tablespan(data = summarized_table,
                 formula = Cylinder:cyl + Engine:vs ~
                   N +
                   (`Horse Power` = Mean:mean_hp + SD:sd_hp) +
                   (`Weight` = Mean:mean_wt + SD:sd_wt),
                 title = "Motor Trend Car Road Tests",
                 subtitle = "A table created with tablespan",
                 footnote = "Data from the infamous mtcars data set.")

tbl |>
  style_subtitle(bold = TRUE) |>
  as_gt()
```

style_title *style_title*

Description

Set the style used for the title of the tablespan table.

Usage

```
style_title(
  tbl,
  background_color = "#ffffff",
  text_color = "#000000",
  font_size = NULL,
  bold = FALSE,
  italic = FALSE,
  openxlsx_style = NULL,
  gt_style = NULL
)
```

Arguments

tbl	tablespan table
background_color	hex code for the background color
text_color	hex code for the text color
font_size	font size
bold	set to TRUE for bold
italic	set to TRUE for italic
openxlsx_style	optional custom openxlsx style. When provided, all other arguments are ignored
gt_style	optional custom gt style. When provided, all other arguments are ignored

Details

The styling for openxlsx and gt works differently:

- openxlsx_style must be a style object created with openxlsx::createStyle. This style will then be applied to the title - gt_style must be a list of gt::tab_style objects to be applied to the table

Value

the tablespan table with added styles

Examples

```
library(tablespan)
library(dplyr)
data("mtcars")

# We want to report the following table:
summarized_table <- mtcars |>
  group_by(cyl, vs) |>
  summarise(N = n(),
            mean_hp = mean(hp),
            sd_hp = sd(hp),
            mean_wt = mean(wt),
            sd_wt = sd(wt))

# Create a tablespan:
tbl <- tablespan(data = summarized_table,
                 formula = Cylinder:cyl + Engine:vs ~
                   N +
                   (`Horse Power` = Mean:mean_hp + SD:sd_hp) +
                   (`Weight` = Mean:mean_wt + SD:sd_wt),
                 title = "Motor Trend Car Road Tests",
                 subtitle = "A table created with tablespan",
                 footnote = "Data from the infamous mtcars data set.")

tbl |>
  style_title(bold = TRUE) |>
  as_gt()
```

`style_vline` *style_vline*

Description

Set the style used for the vertical lines of the tablespan table. Currently only supported for excel export.

Usage

```
style_vline(tbl, openxlsx_style)
```

Arguments

`tbl`	tablespan table
`openxlsx_style`	style used when exporting to openxlsx

Details

- openxlsx_style must be a style object created with openxlsx::createStyle. This style will then be applied to the vertical lines

Value

the tablespan table with added styles

Examples

```
library(tablespan)
library(dplyr)
data("mtcars")

# We want to report the following table:
summarized_table <- mtcars |>
  group_by(cyl, vs) |>
  summarise(N = n(),
            mean_hp = mean(hp),
            sd_hp = sd(hp),
            mean_wt = mean(wt),
            sd_wt = sd(wt))

# Create a tablespan:
tbl <- tablespan(data = summarized_table,
                 formula = Cylinder:cyl + Engine:vs ~
                   N +
                   (`Horse Power` = Mean:mean_hp + SD:sd_hp) +
                   (`Weight` = Mean:mean_wt + SD:sd_wt),
                 title = "Motor Trend Car Road Tests",
                 subtitle = "A table created with tablespan",
                 footnote = "Data from the infamous mtcars data set.")

wb <- tbl |>
  style_vline(
    openxlsx_style = openxlsx::createStyle(
      border = "Top",
      borderColour = "#928505",
      borderStyle = "thin")) |>
  as_excel()
# save workbook to see effect
```

`tablespan` *tablespan*

Description

Create complex table spanners with a simple formula.

Usage

```
tablespan(data, formula, title = NULL, subtitle = NULL, footnote = NULL)
```

Arguments

data	data set
formula	formula to create table
title	string specifying the title of the table
subtitle	string specifying the subtitle of the table
footnote	string specifying the footnote of the table

Details

tablespan provides a formula based approach to adding headers and spanners to an existing data.frame. The objective is to provide a unified, easy to use, but good enough approach to building and exporting tables to Excel, HTML, and LaTeX. To this end, tablespan leverages the awesome packages openxlsx and gt.

Following the tibble approach, tablespan assumes that all items that you may want to use as row names are just columns in your data set (see example). That is, tablespan will allow you to pick some of your items as row names and then just write them in a separate section to the left of the data.

The table headers are defined with a basic formula approach inspired by tables. For example, Species ~ Sepal.Length + Sepal.Width defines a table with Species as the row names and Sepal.Length and Sepal.Width as columns. The output will be similar to the following:

```
|Species | Sepal.Length  Sepal.Width|
|:-------|------------: -----------:|
|setosa  |          5.1          3.5|
|setosa  |          4.9          3.0|
```

Note that the row names (Species) are in a separate block to the left.

You can add spanner labels with as follows:

Species ~ (Sepal = Sepal.Length + Sepal.Width) + (Petal = Sepal.Length + Sepal.Width)

This will result in an output similar to:

```
|        |           Sepal          |          Petal           |
|Species | Sepal.Length| Sepal.Width| Petal.Length| Petal.Width|
|:-------|------------:|-----------:|------------:|-----------:|
|setosa  |          5.1|         3.5|          1.4|         0.2|
```

You can also nest spanners (e.g., Species ~ (Sepal = (Length = Sepal.Length) + (Width = Sepal.Width)).

When exporting tables, you may want to rename some of you columns. For example, you may want to rename Sepal.Length and Petal.Length to Length and Sepal.Width and Petal.Width to Width. With tablespan, you can rename the item in the header using new_name:old_name. For example, Species ~ (Sepal = Length:Sepal.Length + Width:Sepal.Width) + (Petal = Length:Sepal.Length + Width:Sepal.Width) defines a table similar to the following:

```
|        |      Sepal     |      Petal     |
|Species | Length | Width | Length | Width |
|:-------|-------:|------:|-------:|------:|
|setosa  |     5.1|    3.5|     1.4|    0.2|
```

Finally, to create a table without row names, use `1 ~ (Sepal = Length:Sepal.Length + Width:Sepal.Width) + (Petal = Length:Sepal.Length + Width:Sepal.Width)` This defines as table similar to the following:

```
|      Sepal     |      Petal     |
| Length | Width | Length | Width |
|-------:|------:|-------:|------:|
|    5.1|    3.5|     1.4|    0.2|
```

Tables created with `tablespan` can be exported to Excel (using `openxlsx`), HTML (using `gt`), LaTeX (using `gt`), and RTF (using `gt`).

References:

- gt: Iannone R, Cheng J, Schloerke B, Hughes E, Lauer A, Seo J, Brevoort K, Roy O (2024). gt: Easily Create Presentation-Ready Display Tables. R package version 0.11.1.9000, <https://github.com/rstudio/gt>, <https://gt.rstudio.com>.
- tables: Murdoch D (2024). tables: Formula-Driven Table Generation. R package version 0.9.31, <https://dmurdoch.github.io/tables/>.
- openxlsx: Schauberger P, Walker A (2023). _openxlsx: Read, Write and Edit xlsx Files_. R package version 4.2.5.2, <https://ycphs.github.io/openxlsx/>.

Value

Object of class Tablespan with title, subtitle, header info, data, and footnote.

Examples

```
library(tablespan)
library(dplyr)
data("mtcars")

# We want to report the following table:
summarized_table <- mtcars |>
  group_by(cyl, vs) |>
  summarise(N = n(),
            mean_hp = mean(hp),
            sd_hp = sd(hp),
            mean_wt = mean(wt),
            sd_wt = sd(wt))

# Create a tablespan:
tbl <- tablespan(data = summarized_table,
                 formula = Cylinder:cyl + Engine:vs ~
                   N +
                   (`Horse Power` = Mean:mean_hp + SD:sd_hp) +
                   (`Weight` = Mean:mean_wt + SD:sd_wt),
                 title = "Motor Trend Car Road Tests",
                 subtitle = "A table created with tablespan",
                 footnote = "Data from the infamous mtcars data set.")
```

```
tbl

# Add styling:
tbl <- tbl |>
    style_header(background_color = "#000000", text_color = "#ffffff") |>
    style_column(columns = where(is.double), bold = TRUE)

# Export as Excel table:
wb <- as_excel(tbl = tbl)

# Save using openxlsx
# openxlsx::saveWorkbook(wb, "cars.xlsx")

# Export as gt:
gt_tbl <- as_gt(tbl = tbl)
gt_tbl
```

Index