

# Package ‘flextable’

November 12, 2019

**Type** Package

**Title** Functions for Tabular Reporting

**Version** 0.5.6

**Description** Create pretty tables for 'HTML', 'Microsoft Word' and 'Microsoft PowerPoint' documents. Functions are provided to let users create tables, modify and format their content. It extends package 'officer' that does not contain any feature for customized tabular reporting and can be used within R markdown documents.

**License** GPL-3

**LazyData** TRUE

**Imports** stats, utils, grDevices, graphics, officer (>= 0.2.0),  
rmarkdown, knitr, htmltools, xml2, data.table, gdttools (>=  
0.1.6), rlang, base64enc

**RoxygenNote** 6.1.1

**Suggests** testthat, xtable, magrittr, webshot, magick, ggplot2

**Encoding** UTF-8

**URL** <https://davidgohe1.github.io/flextable>

**BugReports** <https://github.com/davidgohe1/flextable/issues>

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** David Gohel [aut, cre],  
Quentin Fazilleau [ctb],  
Maxim Nazarov [ctb] (rmarkdown for docx output),  
Titouan Robert [ctb]

**Maintainer** David Gohel <david.gohel@ardata.fr>

**Repository** CRAN

**Date/Publication** 2019-11-12 18:30:06 UTC

**R topics documented:**

flextable-package . . . . .	4
add_header . . . . .	4
add_header_lines . . . . .	5
add_header_row . . . . .	6
align . . . . .	7
as_b . . . . .	8
as_bracket . . . . .	9
as_chunk . . . . .	9
as_flextable . . . . .	10
as_grouped_data . . . . .	11
as_i . . . . .	12
as_image . . . . .	13
as_paragraph . . . . .	14
as_raster . . . . .	15
as_sub . . . . .	15
as_sup . . . . .	16
autofit . . . . .	17
bg . . . . .	18
body_add_flextable . . . . .	18
bold . . . . .	19
border . . . . .	20
border_inner . . . . .	21
border_inner_h . . . . .	21
border_inner_v . . . . .	22
border_outer . . . . .	23
border_remove . . . . .	24
colformat_char . . . . .	24
colformat_int . . . . .	25
colformat_lgl . . . . .	26
colformat_num . . . . .	27
color . . . . .	28
compose . . . . .	29
delete_part . . . . .	30
dim.flextable . . . . .	30
dim_pretty . . . . .	31
display . . . . .	31
docx_value . . . . .	33
empty_blanks . . . . .	33
fit_to_width . . . . .	34
fix_border_issues . . . . .	35
flextable . . . . .	35
flextable_dim . . . . .	37
font . . . . .	37
fontsize . . . . .	38
footers_flextable_at_bkm . . . . .	39
footnote . . . . .	39

format.flextable . . . . .	40
headers_flextable_at_bkm . . . . .	41
height . . . . .	41
hline . . . . .	42
hline_bottom . . . . .	43
hline_top . . . . .	44
htmltools_value . . . . .	44
hyperlink_text . . . . .	45
italic . . . . .	46
knit_print.flextable . . . . .	47
linerange . . . . .	48
lollipop . . . . .	49
merge_at . . . . .	50
merge_h . . . . .	51
merge_h_range . . . . .	51
merge_none . . . . .	52
merge_v . . . . .	53
minibar . . . . .	54
padding . . . . .	55
ph_with.flextable . . . . .	56
ph_with_flextable . . . . .	57
plot.flextable . . . . .	57
print.flextable . . . . .	58
proc_freq . . . . .	59
rotate . . . . .	60
save_as_html . . . . .	61
save_as_image . . . . .	61
set_caption . . . . .	62
set_formatter . . . . .	63
set_header_footer_df . . . . .	64
set_header_labels . . . . .	65
style . . . . .	66
theme_alafoli . . . . .	67
theme_booktabs . . . . .	67
theme_box . . . . .	68
theme_tron . . . . .	68
theme_tron_legacy . . . . .	69
theme_vader . . . . .	69
theme_vanilla . . . . .	70
theme_zebra . . . . .	71
valign . . . . .	71
vline . . . . .	72
vline_left . . . . .	73
vline_right . . . . .	73
void . . . . .	74
width . . . . .	75
xtable_to_flextable . . . . .	75

---

flextable-package      *flextable: Functions for Tabular Reporting*

---

### Description

The flextable package facilitates access to and manipulation of tabular reporting elements from R. The documentation of functions can be opened with command `help(package = "flextable")`. To learn more about flextable, start with the vignettes: `browseVignettes(package = "flextable")`. `flextable()` function is producing flexible tables where each cell can contain several chunks of text with their own set of formatting properties (bold, font color, etc.). Function `compose` lets customise text of cells.

### See Also

<https://davidgohe1.github.io/flextable/>, `flextable`

---

add\_header      *Add a rows of labels in header or footer part*

---

### Description

Add rows of labels in the flextable's header or footer part. It can be inserted at the top or the bottom of the part. The function is column oriented, labels are specified for each columns, there can be more than a label - resulting in more than a new row.

### Usage

```
add_header(x, top = TRUE, ..., values = NULL)
```

```
add_footer(x, top = TRUE, ..., values = NULL)
```

### Arguments

x	a flextable object
top	should the row be inserted at the top or the bottom.
...	a named list (names are data colnames) of strings specifying corresponding labels to add.
values	a list of name-value pairs of labels or values, names should be existing col_key values. If values is supplied argument ... is ignored.

### Note

when repeating values, they can be merged together with function `merge_h` and `merge_v`.

**See Also**

Other headers and footers: [add\\_header\\_lines](#), [add\\_header\\_row](#), [set\\_header\\_footer\\_df](#)

**Examples**

```
ft <- flextable( head( iris ),
  col_keys = c("Species", "Sepal.Length", "Petal.Length",
              "Sepal.Width", "Petal.Width") )

# start with no header
ft <- delete_part(ft, part = "header")

# add a line of row
ft <- add_header(x = ft, Sepal.Length = "length",
  Sepal.Width = "width", Petal.Length = "length",
  Petal.Width = "width", Species = "Species", top = FALSE )
# add another line of row at the top position
ft <- add_header(ft, Sepal.Length = "Inches",
  Sepal.Width = "Inches", Petal.Length = "Inches",
  Petal.Width = "Inches", top = TRUE )
# merge horizontally when there are identical values
ft <- merge_h(ft, part = "header")

# add a footnote in the footer part
ft <- add_footer(ft, Species = "This is a footnote" )
ft <- merge_at(ft, j = 1:5, part = "footer")

# theme the table
ft <- theme_box(ft)

ft
```

---

add\_header\_lines      *Add a label in a header or footer new row.*

---

**Description**

Add an header or footer new row made of one cell. This is a sugar function to be used when you need to add a title row to a flextable, most of the time it will be used in a context of adding a footnote or adding a title on the top line of the flextable.

**Usage**

```
add_header_lines(x, values = character(0), top = TRUE)
```

```
add_footer_lines(x, values = character(0), top = FALSE)
```

**Arguments**

x	a flextable object
values	a character vector, each element will be added a a new row in the header or footer part.
top	should the row be inserted at the top or the bottom.

**See Also**

Other headers and footers: [add\\_header\\_row](#), [add\\_header](#), [set\\_header\\_footer\\_df](#)

**Examples**

```
ft <- flextable( head( iris ) )
ft <- add_footer_lines(ft, values = "blah blah")
ft <- add_footer_lines(ft, values = c("blah 1", "blah 2"))
autofit(ft)
ft <- flextable( head( iris ) )
ft <- add_header_lines(ft, values = "blah blah")
ft <- add_header_lines(ft, values = c("blah 1", "blah 2"))
autofit(ft)
```

---

add\_header\_row

*Add labels and merge cells in a new header or footer row*

---

**Description**

Add an header or footer new row where some cells are merged, labels are associated with a number of columns to merge. The function is row oriented. One call allow to add one single row.

**Usage**

```
add_header_row(x, top = TRUE, values = character(0),
              colwidths = integer(0))
```

```
add_footer_row(x, top = TRUE, values = character(0),
              colwidths = integer(0))
```

**Arguments**

x	a flextable object
top	should the row be inserted at the top or the bottom.
values	values to add as a character vector
colwidths	the number of columns to merge in the row for each label

**See Also**

Other headers and footers: [add\\_header\\_lines](#), [add\\_header](#), [set\\_header\\_footer\\_df](#)

**Examples**

```
ft <- flextable( head( iris ) )
ft <- add_header_row(ft, values = "blah blah", colwidths = 5)
ft <- add_header_row(ft, values = c("blah", "blah"), colwidths = c(3,2))
ft
ft <- flextable( head( iris ) )
ft <- add_footer_row(ft, values = "blah blah", colwidths = 5)
ft <- add_footer_row(ft, values = c("blah", "blah"), colwidths = c(3,2))
ft
```

align

*Set text alignment***Description**

change text alignment of selected rows and columns of a flextable.

**Usage**

```
align(x, i = NULL, j = NULL, align = "left", part = "body")

align_text_col(x, align = "left", header = TRUE, footer = TRUE)

align_nottext_col(x, align = "right", header = TRUE, footer = TRUE)
```

**Arguments**

x	a flextable object
i	rows selection
j	columns selection
align	text alignment - a single character value, expected value is one of 'left', 'right', 'center', 'justify'.
part	partname of the table (one of 'all', 'body', 'header', 'footer')
header	should the header be aligned with the body
footer	should the footer be aligned with the body

**See Also**

Other sugar functions for table style: [bg](#), [bold](#), [color](#), [empty\\_blanks](#), [fontsize](#), [font](#), [italic](#), [padding](#), [rotate](#), [valign](#)

Other sugar functions for table style: [bg](#), [bold](#), [color](#), [empty\\_blanks](#), [fontsize](#), [font](#), [italic](#), [padding](#), [rotate](#), [valign](#)

## Examples

```
ft <- flextable(mtcars)
ft <- align(ft, align = "center")
ft <- flextable(mtcars)
ft <- align_text_col(ft, align = "left")
ft <- align_nottext_col(ft, align = "right")
ft
```

---

as\_b

*bold chunk*

---

## Description

The function is producing a chunk with bold font.

## Usage

```
as_b(x)
```

## Arguments

x value, if a chunk, the chunk will be updated

## Note

This is a sugar function that ease the composition of complex labels made of different formattings. It should be used inside a call to [as\\_paragraph](#).

## See Also

Other chunk elements for paragraph: [as\\_bracket](#), [as\\_chunk](#), [as\\_image](#), [as\\_i](#), [as\\_sub](#), [as\\_sup](#), [hyperlink\\_text](#), [linerange](#), [lollipop](#), [minibar](#)

## Examples

```
ft <- flextable( head(iris), col_keys = c("dummy") )

ft <- compose(ft, j = "dummy",
  value = as_paragraph(
    as_b(Sepal.Length)
  ) )

autofit(ft)
```



---

as_bracket	<i>chunk with values in brackets</i>
------------	--------------------------------------

---

**Description**

The function is producing a chunk by pasting values and add the result in brackets. It should be used inside a call to [as\\_paragraph](#).

**Usage**

```
as_bracket(..., sep = ", ", p = "(", s = ")")
```

**Arguments**

...	text and column names
sep	separator
p	prefix, default to '('
s	suffix, default to ')'

**See Also**

Other chunk elements for paragraph: [as\\_b](#), [as\\_chunk](#), [as\\_image](#), [as\\_i](#), [as\\_sub](#), [as\\_sup](#), [hyperlink\\_text](#), [linerange](#), [lollipop](#), [minibar](#)

**Examples**

```
ft <- flextable( head(iris),
  col_keys = c("Species", "Sepal", "Petal") )
ft <- set_header_labels(ft, Sepal="Sepal", Petal="Petal")
ft <- compose(ft, j = "Sepal",
  value = as_paragraph( as_bracket(Sepal.Length, Sepal.Width) ) )
ft <- compose(ft, j = "Petal",
  value = as_paragraph( as_bracket(Petal.Length, Petal.Width) ) )
autofit(ft)
```

---

as_chunk	<i>chunk of text wrapper</i>
----------	------------------------------

---

**Description**

The function lets add text within flextable objects with function [compose](#). It should be used inside a call to [as\\_paragraph](#).

**Usage**

```
as_chunk(x, props = NULL, formater = format_fun, ...)
```

**Arguments**

x	text or any element that can be formatted as text with function provided in argument formater.
props	an <code>fp_text</code> object to be used to format the text. If not specified, it will be the default value corresponding to the cell.
formater	a function that will format x as a character vector.
...	additional arguments for formater function.

**See Also**

Other chunk elements for paragraph: [as\\_bracket](#), [as\\_b](#), [as\\_image](#), [as\\_i](#), [as\\_sub](#), [as\\_sup](#), [hyperlink\\_text](#), [linerange](#), [lollipop](#), [minibar](#)

**Examples**

```
library(officer)

myft <- flextable( head(iris))

myft <- compose( myft, j = "Sepal.Length",
  value = as_paragraph(
    "Sepal.Length value is ",
    as_chunk(Sepal.Length, props = fp_text(color = "red"))
  ),
  part = "body")
myft <- color(myft, color = "gray40", part = "all")
autofit(myft)
```

---

as\_flextable                      *method to convert object to flextable*

---

**Description**

This is a convenient function to let users create flextable bindings from any objects.

**Usage**

```
as_flextable(x, ...)

## S3 method for class 'grouped_data'
as_flextable(x, col_keys = NULL,
  hide_grouplabel = FALSE, ...)
```

**Arguments**

x                    object to be transformed as flextable  
 ...                  arguments for custom methods  
 col\_keys            columns names/keys to display. If some column names are not in the dataset, they will be added as blank columns by default.  
 hide\_grouplabel    if TRUE, group label will not be rendered, only level/value will be rendered.

**Examples**

```
# as_flextable and as_grouped_data -----
if( require("magrittr") ){
  library(data.table)
  C02 <- C02
  setDT(C02)
  C02$conc <- as.integer(C02$conc)

  data_co2 <- dcast(C02, Treatment + conc ~ Type,
                   value.var = "uptake", fun.aggregate = mean)
  data_co2 <- as_grouped_data(x = data_co2, groups = c("Treatment"))

  zz <- as_flextable( data_co2 ) %>%
    add_footer_lines("dataset C02 has been used for this flextable") %>%
    add_header_lines("mean of carbon dioxide uptake in grass plants") %>%
    set_header_labels(conc = "Concentration") %>%
    autofit() %>%
    width(width = c(1, 1, 1))
  zz
}
```

---

as\_grouped\_data            *grouped data transformation*

---

**Description**

Repeated consecutive values of group columns will be used to define the title of the groups and will be added as a row title.

**Usage**

```
as_grouped_data(x, groups, columns = NULL)
```

**Arguments**

x                    dataset  
 groups              columns names to be used as row separators.  
 columns             columns names to keep

**See Also**

[as\\_flextable](#)

**Examples**

```
# as_grouped_data -----
library(data.table)
C02 <- C02
setDT(C02)
C02$conc <- as.integer(C02$conc)

data_co2 <- dcast(C02, Treatment + conc ~ Type,
  value.var = "uptake", fun.aggregate = mean)
data_co2
data_co2 <- as_grouped_data(x = data_co2, groups = c("Treatment"))
data_co2
```

---

as\_i

*italic chunk*

---

**Description**

The function is producing a chunk with italic font.

**Usage**

```
as_i(x)
```

**Arguments**

x value, if a chunk, the chunk will be updated

**Note**

This is a sugar function that ease the composition of complex labels made of different formattings. It should be used inside a call to [as\\_paragraph](#).

**See Also**

Other chunk elements for paragraph: [as\\_bracket](#), [as\\_b](#), [as\\_chunk](#), [as\\_image](#), [as\\_sub](#), [as\\_sup](#), [hyperlink\\_text](#), [linerange](#), [lollipop](#), [minibar](#)

**Examples**

```
ft <- flextable( head(iris), col_keys = c("dummy") )

ft <- compose(ft, j = "dummy",
  value = as_paragraph(
    as_i(Sepal.Length)
  ) )

autofit(ft)
```

as\_image

*image chunk wrapper***Description**

The function lets add images within flextable objects with function [compose](#). It should be used inside a call to [as\\_paragraph](#).

**Usage**

```
as_image(src, width = 0.5, height = 0.2, ...)
```

**Arguments**

src	image filename
width, height	size of the png file in inches
...	unused argument

**Note**

PowerPoint cannot mix images and text in a paragraph, images are removed when outputting to PowerPoint format.

**See Also**

[compose](#), [as\\_paragraph](#)

Other chunk elements for paragraph: [as\\_bracket](#), [as\\_b](#), [as\\_chunk](#), [as\\_i](#), [as\\_sub](#), [as\\_sup](#), [hyperlink\\_text](#), [linerange](#), [lollipop](#), [minibar](#)

**Examples**

```
img.file <- file.path( R.home("doc"), "html", "logo.jpg" )
library(officer)

myft <- flextable( head(iris))

myft <- compose( myft, i = 1:3, j = 1,
  value = as_paragraph(
```

```

as_image(src = img.file, width = .20, height = .15),
" blah blah ",
as_chunk(Sepal.Length, props = fp_text(color = "red"))
),
part = "body")

autofit(myft)

```

---

as\_paragraph

*concatenate chunks in a flextable*


---

### Description

The function is concatenating text and images within paragraphs of a flextable object, this function is to be used with function [compose](#).

### Usage

```
as_paragraph(..., list_values = NULL)
```

### Arguments

...	chunk elements that are defining paragraph
list_values	a list of chunk elements that are defining paragraph. If specified argument ... is unused.

### See Also

[as\\_chunk](#), [minibar](#), [as\\_image](#), [hyperlink\\_text](#)

### Examples

```

library(officer)
myft <- flextable( head(iris, n = 10 ))

myft <- compose( myft, j = 1,
  value = as_paragraph(
    minibar(value = Sepal.Length, max = max(Sepal.Length)),
    " ",
    as_chunk( Sepal.Length, formater = formatC,
      props = fp_text(color = "orange") ),
    " blah blah"
  ),
  part = "body")

autofit(myft)

```

---

as_raster	<i>get a flextable as a raster</i>
-----------	------------------------------------

---

**Description**

save a flextable as an image and return the corresponding raster. This function has been implemented to let flextable be printed on a ggplot object.

**Usage**

```
as_raster(x, zoom = 2, expand = 2)
```

**Arguments**

x                    a flextable object  
zoom, expand        parameters used by webshot function.

**Note**

This function requires packages: webshot and magick.

**See Also**

Other flextable print function: [docx\\_value](#), [format.flextable](#), [htmltools\\_value](#), [knit\\_print.flextable](#), [plot.flextable](#), [print.flextable](#), [save\\_as\\_html](#), [save\\_as\\_image](#)

**Examples**

```
ft <- qflextable( head( mtcars ) )  
## Not run:  
if( require("ggplot2") && require("webshot") ){  
  print(qplot(speed, dist, data = cars, geom = "point"))  
  grid::grid.raster(as_raster(ft))  
}  
  
## End(Not run)
```

---

as_sub	<i>subscript chunk</i>
--------	------------------------

---

**Description**

The function is producing a chunk with subscript vertical alignment.

**Usage**

```
as_sub(x)
```

**Arguments**

x value, if a chunk, the chunk will be updated

**Note**

This is a sugar function that ease the composition of complex labels made of different formattings. It should be used inside a call to [as\\_paragraph](#).

**See Also**

Other chunk elements for paragraph: [as\\_bracket](#), [as\\_b](#), [as\\_chunk](#), [as\\_image](#), [as\\_i](#), [as\\_sup](#), [hyperlink\\_text](#), [linerange](#), [lollipop](#), [minibar](#)

**Examples**

```
ft <- flextable( head(iris), col_keys = c("dummy") )

ft <- compose(ft, i = 1, j = "dummy", part = "header",
  value = as_paragraph(
    as_sub("Sepal.Length"),
    " anything "
  ) )

autofit(ft)
```

---

as\_sup

*superscript chunk*


---

**Description**

The function is producing a chunk with superscript vertical alignment.

**Usage**

```
as_sup(x)
```

**Arguments**

x value, if a chunk, the chunk will be updated

**Note**

This is a sugar function that ease the composition of complex labels made of different formattings. It should be used inside a call to [as\\_paragraph](#).

**See Also**

Other chunk elements for paragraph: [as\\_bracket](#), [as\\_b](#), [as\\_chunk](#), [as\\_image](#), [as\\_i](#), [as\\_sub](#), [hyperlink\\_text](#), [linerange](#), [lollipop](#), [minibar](#)



## Examples

```
ft <- flextable( head(iris), col_keys = c("dummy") )

ft <- compose(ft, i = 1, j = "dummy", part = "header",
  value = as_paragraph(
    " anything ",
    as_sup("Sepal.Width")
  ) )

autofit(ft)
```

---

autofit

*Adjusts cell widths and heights*

---

## Description

compute and apply optimized widths and heights. This function is to be used when the table widths and heights should automatically be adjusted to fit the size of the content.

## Usage

```
autofit(x, add_w = 0.1, add_h = 0.1)
```

## Arguments

x	flextable object
add_w	extra width to add in inches
add_h	extra height to add in inches

## See Also

Other flextable dimensions: [dim.flextable](#), [dim\\_pretty](#), [fit\\_to\\_width](#), [flextable\\_dim](#), [height](#), [width](#)

## Examples

```
ft <- flextable(mtcars)
ft <- autofit(ft)
ft
```

---

bg	<i>Set background color</i>
----	-----------------------------

---

**Description**

change background color of selected rows and columns of a flextable.

**Usage**

```
bg(x, i = NULL, j = NULL, bg, part = "body")
```

**Arguments**

x	a flextable object
i	rows selection
j	columns selection
bg	color to use as background color
part	partname of the table (one of 'all', 'body', 'header', 'footer')

**See Also**

Other sugar functions for table style: [align](#), [bold](#), [color](#), [empty\\_blanks](#), [fontsize](#), [font](#), [italic](#), [padding](#), [rotate](#), [valign](#)

**Examples**

```
ft <- flextable(mtcars)
ft <- bg(ft, bg = "#DDDDDD", part = "header")
```

---

body_add_flextable	<i>add flextable into a Word document</i>
--------------------	---

---

**Description**

add a flextable into a Word document.

**Usage**

```
body_add_flextable(x, value, align = "center", pos = "after",
  split = FALSE)

body_replace_flextable_at_bkm(x, bookmark, value, align = "center",
  split = FALSE)
```

**Arguments**

x	an rdocx object
value	flextable object
align	left, center (default) or right.
pos	where to add the flextable relative to the cursor, one of "after", "before", "on" (end of line).
split	set to TRUE if you want to activate Word option 'Allow row to break across pages'.
bookmark	bookmark id

**body\_replace\_flextable\_at\_bkm**

Use this function if you want to replace a paragraph containing a bookmark with a flextable. As a side effect, the bookmark will be lost.

**Examples**

```
library(officer)
ft <- flextable(head(mtcars))
ft <- theme_zebra(ft)
ft <- autofit(ft)
doc <- read_docx()
doc <- body_add_flextable(doc, value = ft)
fileout <- tempfile(fileext = ".docx")
# fileout <- "test.docx" # uncomment to write in your working directory
print(doc, target = fileout)
```

---

bold	<i>Set bold font</i>
------	----------------------

---

**Description**

change font weight of selected rows and columns of a flextable.

**Usage**

```
bold(x, i = NULL, j = NULL, bold = TRUE, part = "body")
```

**Arguments**

x	a flextable object
i	rows selection
j	columns selection
bold	boolean value
part	partname of the table (one of 'all', 'body', 'header', 'footer')

**See Also**

Other sugar functions for table style: [align](#), [bg](#), [color](#), [empty\\_blanks](#), [fontsize](#), [font](#), [italic](#), [padding](#), [rotate](#), [valign](#)

**Examples**

```
ft <- flextable(mtcars)
ft <- bold(ft, bold = TRUE, part = "header")
```

---

border	<i>Set cell borders</i>
--------	-------------------------

---

**Description**

change borders of selected rows and columns of a flextable.

**Usage**

```
border(x, i = NULL, j = NULL, border = NULL, border.top = NULL,
       border.bottom = NULL, border.left = NULL, border.right = NULL,
       part = "body")
```

**Arguments**

x	a flextable object
i	rows selection
j	columns selection
border	border (shortcut for top, bottom, left and right)
border.top	border top
border.bottom	border bottom
border.left	border left
border.right	border right
part	partname of the table (one of 'all', 'body', 'header', 'footer')

**Note**

this function requires careful settings to avoid overlapping borders.

**See Also**

Other borders management: [border\\_inner\\_h](#), [border\\_inner\\_v](#), [border\\_inner](#), [border\\_outer](#), [border\\_remove](#), [hline\\_bottom](#), [hline\\_top](#), [hline](#), [vline\\_left](#), [vline\\_right](#), [vline](#)

**Examples**

```
library(officer)
ft <- flextable(mtcars)
ft <- border(ft, border.top = fp_border(color = "orange") )
```

---

border_inner	<i>set vertical &amp; horizontal inner borders</i>
--------------	--

---

**Description**

The function is applying a vertical and horizontal borders to inner content of one or all parts of a flextable.

**Usage**

```
border_inner(x, border = NULL, part = "all")
```

**Arguments**

x	a flextable object
border	border defined by a call to <a href="#">fp_border</a>
part	partname of the table (one of 'all', 'body', 'header', 'footer')

**See Also**

Other borders management: [border\\_inner\\_h](#), [border\\_inner\\_v](#), [border\\_outer](#), [border\\_remove](#), [border](#), [hline\\_bottom](#), [hline\\_top](#), [hline](#), [vline\\_left](#), [vline\\_right](#), [vline](#)

**Examples**

```
library(officer)
std_border = fp_border(color="orange", width = 1)

dat <- iris[c(1:5, 51:55, 101:105),]
ft <- flextable(dat)
ft <- border_remove(x = ft)

# add inner vertical borders
ft <- border_inner(ft, border = std_border )
ft
```

---

border_inner_h	<i>set inner borders</i>
----------------	--------------------------

---

**Description**

The function is applying a border to inner content of one or all parts of a flextable.

**Usage**

```
border_inner_h(x, border = NULL, part = "body")
```

**Arguments**

x	a flextable object
border	border defined by a call to <a href="#">fp_border</a>
part	partname of the table (one of 'all', 'body', 'header', 'footer')

**See Also**

Other borders management: [border\\_inner\\_v](#), [border\\_inner](#), [border\\_outer](#), [border\\_remove](#), [border](#), [hline\\_bottom](#), [hline\\_top](#), [hline](#), [vline\\_left](#), [vline\\_right](#), [vline](#)

**Examples**

```
library(officer)
std_border = fp_border(color="orange", width = 1)

dat <- iris[c(1:5, 51:55, 101:105),]
ft <- flextable(dat)
ft <- border_remove(x = ft)

# add inner horizontal borders
ft <- border_inner_h(ft, border = std_border )
ft
```

---

border\_inner\_v      *set vertical inner borders*

---

**Description**

The function is applying a vertical border to inner content of one or all parts of a flextable.

**Usage**

```
border_inner_v(x, border = NULL, part = "all")
```

**Arguments**

x	a flextable object
border	border defined by a call to <a href="#">fp_border</a>
part	partname of the table (one of 'all', 'body', 'header', 'footer')

**See Also**

Other borders management: [border\\_inner\\_h](#), [border\\_inner](#), [border\\_outer](#), [border\\_remove](#), [border](#), [hline\\_bottom](#), [hline\\_top](#), [hline](#), [vline\\_left](#), [vline\\_right](#), [vline](#)

**Examples**

```

library(officer)
std_border = fp_border(color="orange", width = 1)

dat <- iris[c(1:5, 51:55, 101:105),]
ft <- flextable(dat)
ft <- border_remove(x = ft)

# add inner vertical borders
ft <- border_inner_v(ft, border = std_border )
ft

```

---

border_outer	<i>set outer borders</i>
--------------	--------------------------

---

**Description**

The function is applying a border to outer cells of one or all parts of a flextable.

**Usage**

```
border_outer(x, border = NULL, part = "all")
```

**Arguments**

x	a flextable object
border	border defined by a call to <a href="#">fp_border</a>
part	partname of the table (one of 'all', 'body', 'header', 'footer')

**See Also**

Other borders management: [border\\_inner\\_h](#), [border\\_inner\\_v](#), [border\\_inner](#), [border\\_remove](#), [border](#), [hline\\_bottom](#), [hline\\_top](#), [hline](#), [vline\\_left](#), [vline\\_right](#), [vline](#)

**Examples**

```

library(officer)
big_border = fp_border(color="red", width = 2)

dat <- iris[c(1:5, 51:55, 101:105),]
ft <- flextable(dat)
ft <- border_remove(x = ft)

# add outer borders
ft <- border_outer(ft, part="all", border = big_border )
ft

```

---

border_remove	<i>remove borders</i>
---------------	-----------------------

---

### Description

The function is deleting all borders of the flextable object.

### Usage

```
border_remove(x)
```

### Arguments

x                    a flextable object

### See Also

Other borders management: [border\\_inner\\_h](#), [border\\_inner\\_v](#), [border\\_inner](#), [border\\_outer](#), [border](#), [hline\\_bottom](#), [hline\\_top](#), [hline](#), [vline\\_left](#), [vline\\_right](#), [vline](#)

### Examples

```
dat <- iris[c(1:5, 51:55, 101:105),]
ft <- flextable(dat)
ft <- theme_box(ft)
ft

# remove all borders
ft <- border_remove(x = ft)
ft
```

---

colformat_char	<i>format character cells</i>
----------------	-------------------------------

---

### Description

Format character cells in a flextable.

### Usage

```
colformat_char(x, ...)

## S3 method for class 'flextable'
colformat_char(x, j = NULL, col_keys = NULL,
  na_str = "", prefix = "", suffix = "", ...)
```



**Arguments**

x a flextable object  
 ... additional arguments, i can be used to specify a row selector.  
 j columns selection.  
 col\_keys names of the colkeys. Will be deprecated in favor of j in the next version.  
 na\_str string to be used for NA values  
 prefix, suffix string to be used as prefix or suffix

**See Also**

Other cells formatters: [colformat\\_int](#), [colformat\\_lgl](#), [colformat\\_num](#), [compose](#)

**Examples**

```

dat <- iris
ft <- flextable(dat)
ft <- colformat_char(
  x = ft, col_keys = "Species", suffix = "!")
autofit(ft)

```

---

colformat\_int      *format integer cells*

---

**Description**

Format integer cells in a flextable.

**Usage**

```

colformat_int(x, ...)

## S3 method for class 'flextable'
colformat_int(x, j = NULL, col_keys = NULL,
  big.mark = ",", na_str = "", prefix = "", suffix = "", ...)

```

**Arguments**

x a flextable object  
 ... additional arguments, i can be used to specify a row selector.  
 j columns selection.  
 col\_keys names of the colkeys. Will be deprecated in favor of j in the next version.  
 big.mark see [formatC](#)  
 na\_str string to be used for NA values  
 prefix string to be used as prefix or suffix  
 suffix string to be used as prefix or suffix

**See Also**

Other cells formatters: [colformat\\_char](#), [colformat\\_lgl](#), [colformat\\_num](#), [compose](#)

**Examples**

```
dat <- mtcars

ft <- flextable(dat)
colkeys <- c("vs", "am", "gear", "carb")
ft <- colformat_int(x = ft, col_keys = colkeys, prefix = "# ")
autofit(ft)
```

---

<code>colformat_lgl</code>	<i>format logical cells</i>
----------------------------	-----------------------------

---

**Description**

Format logical cells in a flextable.

**Usage**

```
colformat_lgl(x, ...)

## S3 method for class 'flextable'
colformat_lgl(x, j = NULL, col_keys = NULL,
  true = "true", false = "false", na_str = "", prefix = "",
  suffix = "", ...)
```

**Arguments**

<code>x</code>	a flextable object
<code>...</code>	additional arguments, <code>i</code> can be used to specify a row selector.
<code>j</code>	columns selection.
<code>col_keys</code>	names of the colkeys. Will be deprecated in favor of <code>j</code> in the next version.
<code>false, true</code>	string to be used for logical
<code>na_str</code>	string to be used for NA values
<code>prefix</code>	string to be used as prefix or suffix
<code>suffix</code>	string to be used as prefix or suffix

**See Also**

Other cells formatters: [colformat\\_char](#), [colformat\\_int](#), [colformat\\_num](#), [compose](#)

**Examples**

```
dat <- data.frame(a = c(TRUE, FALSE), b = c(FALSE, TRUE))

ft <- flextable(dat)
ft <- colformat_lgl(x = ft, col_keys = c("a", "b"))
autofit(ft)
```

---

colformat_num	<i>format numeric cells</i>
---------------	-----------------------------

---

**Description**

Format numeric cells in a flextable.

**Usage**

```
colformat_num(x, ...)

## S3 method for class 'flextable'
colformat_num(x, j = NULL, col_keys = NULL,
  big.mark = ",", digits = 2, na_str = "", prefix = "",
  suffix = "", ...)
```

**Arguments**

x	a flextable object
...	additional arguments, i can be used to specify a row selector.
j	columns selection.
col_keys	names of the colkeys. Will be deprecated in favor of j in the next version.
big.mark, digits	see <a href="#">formatC</a>
na_str	string to be used for NA values
prefix	string to be used as prefix or suffix
suffix	string to be used as prefix or suffix

**See Also**

Other cells formatters: [colformat\\_char](#), [colformat\\_int](#), [colformat\\_lgl](#), [compose](#)

**Examples**

```

dat <- iris
dat[1:4, 1] <- NA
dat[, 2] <- dat[, 2] * 1000000

ft <- flextable(dat)
colkeys = c("Sepal.Length", "Sepal.Width",
            "Petal.Length", "Petal.Width")
ft <- colformat_num(
  x = ft, col_keys = colkeys,
  big.mark="," , digits = 2, na_str = "N/A")
autofit(ft)

```

---

color	<i>Set font color</i>
-------	-----------------------

---

**Description**

change font color of selected rows and columns of a flextable.

**Usage**

```
color(x, i = NULL, j = NULL, color, part = "body")
```

**Arguments**

x	a flextable object
i	rows selection
j	columns selection
color	color to use as font color
part	partname of the table (one of 'all', 'body', 'header', 'footer')

**See Also**

Other sugar functions for table style: [align](#), [bg](#), [bold](#), [empty\\_blanks](#), [fontsize](#), [font](#), [italic](#), [padding](#), [rotate](#), [valign](#)

**Examples**

```

ft <- flextable(mtcars)
ft <- color(ft, color = "orange", part = "header")

```

---

compose	<i>Define flextable displayed values</i>
---------	--

---

## Description

Modify flextable displayed values. Function is handling complex formatting as well as image insertion.

## Usage

```
compose(x, i = NULL, j = NULL, value, part = "body")
```

```
mk_par(x, i = NULL, j = NULL, value, part = "body")
```

## Arguments

x	a flextable object
i	rows selection
j	column selection
value	a call to function <a href="#">as_paragraph</a> .
part	partname of the table (one of 'all', 'body', 'header', 'footer')

## mk\_par

Function `mk_par` is another name for `compose` as there is an unwanted conflict with package `purrr`.

## See Also

Other cells formatters: [colformat\\_char](#), [colformat\\_int](#), [colformat\\_lgl](#), [colformat\\_num](#)

## Examples

```
library(officer)
ft <- flextable(head( mtcars, n = 10))
ft <- compose(ft, j = "carb", i = ~ drat > 3.5,
  value = as_paragraph("carb is ", as_chunk( sprintf("%.1f", carb)) )
)
ft <- autofit(ft)
```

delete\_part                    *delete flextable part*

---

**Description**

indicate to not print a part of the flextable, i.e. an header, footer or the body.

**Usage**

```
delete_part(x, part = "header")
```

**Arguments**

x                                a flextable object  
part                            partname of the table to delete (one of 'body', 'header' or 'footer').

**Examples**

```
ft <- flextable( head( iris ) )  
ft <- delete_part(x = ft, part = "header")  
ft
```

---

dim.flextable                    *Get widths and heights of flextable*

---

**Description**

returns widths and heights for each table columns and rows. Values are expressed in inches.

**Usage**

```
## S3 method for class 'flextable'  
dim(x)
```

**Arguments**

x                                flextable object

**See Also**

Other flextable dimensions: [autofit](#), [dim\\_pretty](#), [fit\\_to\\_width](#), [flextable\\_dim](#), [height](#), [width](#)

**Examples**

```
ft <- flextable(head(iris))  
dim(ft)
```

---

dim_pretty	<i>Calculate pretty dimensions</i>
------------	------------------------------------

---

**Description**

return minimum estimated widths and heights for each table columns and rows in inches.

**Usage**

```
dim_pretty(x, part = "all")
```

**Arguments**

x	flextable object
part	partname of the table (one of 'all', 'body', 'header' or 'footer')

**See Also**

Other flextable dimensions: [autofit](#), [dim.flextable](#), [fit\\_to\\_width](#), [flextable\\_dim](#), [height](#), [width](#)

**Examples**

```
ft <- flextable(mtcars)
dim_pretty(ft)
```

---

display	<i>Define flextable displayed values</i>
---------	--

---

**Description**

Modify flextable displayed values by specifying a string expression. Function is handling complex formatting as well as image insertion.

**Usage**

```
display(x, i = NULL, col_key, pattern, formatters = list(),
        fprops = list(), part = "body")
```

**Arguments**

x	a flextable object
i	rows selection
col_key	column to modify, a single character
pattern	string to format
formatters	a list of formula, left side for the name, right side for the content.
fprops	a named list of <code>fp_text</code>
part	partname of the table (one of 'all', 'body', 'header', 'footer')

**pattern**

It defined the template used to format the produced strings. Names enclosed by double braces will be evaluated as R code, the corresponding R code is defined with the argument `formatters`.

**formatters**

Each compound is specifying the R code to execute to produce strings that will be substituted in the `pattern` argument. An element must be a formula: the left-hand side is a name (matching a name enclosed by double braces in `pattern`) and the right-hand side is an R expression to be evaluated (that will produce the corresponding strings).

The function is designed to work with columns in the dataset provided to `flextable` (the `col_keys`).

**fprops**

A named list of `fp_text`. It defines the formatting properties associated to a compound in `formatters`. If not defined for an element of `formatters`, the default formatting properties will be applied.

**Note**

You should use `compose` instead - the function is easier to use. Function `display` will be deprecated in the next release.

**Examples**

```
library(officer)
# Formatting data values example -----
ft <- flextable(head(mtcars, n = 10))
ft <- display(ft, col_key = "carb",
  i = ~ drat > 3.5, pattern = "# {{carb}}",
  formatters = list(carb ~ sprintf("%.1f", carb)),
  fprops = list(carb = fp_text(color="orange") ) )
ft <- autofit(ft)
```



---

docx_value	<i>flextable docx string</i>
------------	------------------------------

---

### Description

get openxml raw code for Word from a flextable object.

### Usage

```
docx_value(x, print = TRUE)
```

### Arguments

x	a flextable object
print	print output if TRUE

### See Also

Other flextable print function: [as\\_raster](#), [format.flextable](#), [htmltools\\_value](#), [knit\\_print.flextable](#), [plot.flextable](#), [print.flextable](#), [save\\_as\\_html](#), [save\\_as\\_image](#)

### Examples

```
docx_value(flextable(iris[1:5,]))
```

---

empty_blanks	<i>make blank columns as transparent</i>
--------------	--

---

### Description

blank columns are set as transparent. This is a shortcut function that will delete top and bottom borders, change background color to transparent and display empty content.

### Usage

```
empty_blanks(x)
```

### Arguments

x	a flextable object
---	--------------------

### See Also

Other sugar functions for table style: [align](#), [bg](#), [bold](#), [color](#), [fontsize](#), [font](#), [italic](#), [padding](#), [rotate](#), [valign](#)

**Examples**

```

typology <- data.frame(
  col_keys = c( "Sepal.Length", "Sepal.Width", "Petal.Length",
               "Petal.Width", "Species" ),
  what = c("Sepal", "Sepal", "Petal", "Petal", " " ),
  measure = c("Length", "Width", "Length", "Width", "Species"),
  stringsAsFactors = FALSE )
typology

ft <- flextable(head(iris), col_keys = c("Species",
    "break1", "Sepal.Length", "Sepal.Width",
    "break2", "Petal.Length", "Petal.Width") )
ft <- set_header_df(ft, mapping = typology, key = "col_keys" )
ft <- merge_h(ft, part = "header")
ft <- theme_vanilla(ft)
ft <- empty_blanks(ft)
ft <- width(ft, j = c(2, 5), width = .1 )
ft

```

---

fit\_to\_width

*fit a flextable to a maximum width*


---

**Description**

decrease font size for each cell incrementally until it fits a given `max_width`.

**Usage**

```
fit_to_width(x, max_width, inc = 1L, max_iter = 20)
```

**Arguments**

<code>x</code>	flextable object
<code>max_width</code>	maximum width to fit in inches
<code>inc</code>	the font size decrease for each step
<code>max_iter</code>	maximum iterations

**See Also**

Other flextable dimensions: [autofit](#), [dim.flextable](#), [dim\\_pretty](#), [flextable\\_dim](#), [height](#), [width](#)

**Examples**

```

ft <- qflextable(head(mtcars))
ft <- padding(ft, padding = 0, part = "all")
fit_to_width(ft, max_width = 6)

```

---

fix_border_issues	<i>fix border issues when cell are merged</i>
-------------------	---

---

### Description

When cells are merged, the rendered borders will be those of the first cell. If a column is made of three merged cells, the bottom border that will be seen will be the bottom border of the first cell in the column. From a user point of view, this is wrong, the bottom should be the one defined for cell 3. This function modify the border values to avoid that effect.

### Usage

```
fix_border_issues(x, part = "all")
```

### Arguments

x	flextable object
part	partname of the table (one of 'all', 'body', 'header', 'footer')

### Examples

```
if( require(magrittr) ){
  library(officer)
  ft <- data.frame(a = 1:5, b = 6:10) %>%
    flextable() %>%
    theme_box() %>%
    merge_at(i = 4:5, j = 1, part = "body") %>%
    hline(i = 5, part = "body",
         border = fp_border(color = "red", width = 5) )
  print(ft)
  fix_border_issues(ft) %>% print()
}
```

---

flextable	<i>flextable creation</i>
-----------	---------------------------

---

### Description

Create a flextable object with function `flextable`.

`flextable` are designed to make tabular reporting easier for R users. Functions are available to let you format text, paragraphs and cells; table cells can be merge vertically or horizontally, row headers can easilly be defined, rows heights and columns widths can be manually set or automatically computed.

## Usage

```
flextable(data, col_keys = names(data), cwidth = 0.75,  
  cheight = 0.25, defaults = list(), theme_fun = theme_booktabs)
```

```
qflextable(data)
```

```
regulartable(data, col_keys = names(data), cwidth = 0.75,  
  cheight = 0.25)
```

## Arguments

<code>data</code>	dataset
<code>col_keys</code>	columns names/keys to display. If some column names are not in the dataset, they will be added as blank columns by default.
<code>cwidth, cheight</code>	initial width and height to use for cell sizes in inches.
<code>defaults</code>	a list of default values for formats, supported options are <code>fontname</code> , <code>font.size</code> , <code>color</code> and <code>padding</code> .
<code>theme_fun</code>	a function theme to apply before returning the flextable. set to <code>NULL</code> for none.

## Details

A flextable is made of 3 parts: header, body and footer.

Most functions have an argument named `part` that will be used to specify what part of the table should be modified.

## qflextable

`qflextable` is a convenient tool to produce quickly a flextable for reporting

## Note

Function `regulartable` is maintained for compatibility with old codes made by users but be aware it produces the same exact object than `flextable`.

## Examples

```
ft <- flextable(mtcars)  
ft
```

---

flextable_dim	<i>width and height of a flextable object</i>
---------------	---

---

### Description

Returns the width, height and aspect ratio of a flextable in a named list. The width and height are in inches. The aspect ratio is the ratio corresponding to height/width.

Names of the list are width, height and aspect\_ratio.

### Usage

```
flextable_dim(x)
```

### Arguments

x                    a flextable object

### See Also

Other flextable dimensions: [autofit](#), [dim.flextable](#), [dim.pretty](#), [fit\\_to\\_width](#), [height](#), [width](#)

### Examples

```
ft <- flextable(head(iris))
flextable_dim(ft)
ft <- autofit(ft)
flextable_dim(ft)
```

---

font	<i>Set font</i>
------	-----------------

---

### Description

change font of selected rows and columns of a flextable.

### Usage

```
font(x, i = NULL, j = NULL, fontname, part = "body")
```

### Arguments

x                    a flextable object  
i                    rows selection  
j                    columns selection  
fontname            string value, the font name.  
part                partname of the table (one of 'all', 'body', 'header', 'footer')

**See Also**

Other sugar functions for table style: [align](#), [bg](#), [bold](#), [color](#), [empty\\_blanks](#), [fontsize](#), [italic](#), [padding](#), [rotate](#), [valign](#)

**Examples**

```
require("gdtools")
fontname <- "Times"

if( !font_family_exists(fontname) ){
  ft <- flextable(head(iris))
  ft <- font(ft, fontname = fontname, part = "header")
}
```

---

 fontsize

*Set font size*


---

**Description**

change font size of selected rows and columns of a flextable.

**Usage**

```
fontsize(x, i = NULL, j = NULL, size = 11, part = "body")
```

**Arguments**

x	a flextable object
i	rows selection
j	columns selection
size	integer value (points)
part	partname of the table (one of 'all', 'body', 'header', 'footer')

**See Also**

Other sugar functions for table style: [align](#), [bg](#), [bold](#), [color](#), [empty\\_blanks](#), [font](#), [italic](#), [padding](#), [rotate](#), [valign](#)

**Examples**

```
ft <- flextable(mtcars)
ft <- fontsize(ft, size = 14, part = "header")
```

---

 footers\_flextable\_at\_bkm

*add flextable at a bookmark location in document's footer*


---

### Description

replace in the footer of a document a paragraph containing a bookmark by a flextable. A bookmark will be considered as valid if enclosing words within a paragraph; i.e., a bookmark along two or more paragraphs is invalid, a bookmark set on a whole paragraph is also invalid, but bookmarking few words inside a paragraph is valid.

### Usage

```
footers_flextable_at_bkm(x, bookmark, value)
```

### Arguments

x	an rdocx object
bookmark	bookmark id
value	a flextable object

---

 footnote

*add footnotes to flextable*


---

### Description

add footnotes to a flextable object. A symbol is appened where the footnote is defined and the note is appened in the footer part of the table.

### Usage

```
footnote(x, i = NULL, j = NULL, value, ref_symbols = NULL,
  part = "body")
```

### Arguments

x	a flextable object
i	rows selection
j	column selection
value	a call to function <a href="#">as_paragraph</a> .
ref_symbols	character value, symbols to append that will be used as references to notes.
part	partname of the table (one of 'body', 'header', 'footer')

**Examples**

```
ft <- flextable(head(iris))
ft <- footnote( ft, i = 1, j = 1:3,
  value = as_paragraph(
    c("This is footnote one",
      "This is footnote two",
      "This is footnote three")
  ),
  ref_symbols = c("a", "b", "c"),
  part = "header")
ft <- valign(ft, valign = "bottom", part = "header")
ft <- autofit(ft)
```

---

format.flextable	<i>Encode flextable in a document format.</i>
------------------	---

---

**Description**

Encode flextable in a document format, html, docx, pptx.

This function is exported so that users can create their own custom component.

**Usage**

```
## S3 method for class 'flextable'
format(x, type, ...)
```

**Arguments**

x	flextable object
type	one of pptx, docx or html.
...	unused

**See Also**

Other flextable print function: [as\\_raster](#), [docx\\_value](#), [htmltools\\_value](#), [knit\\_print.flextable](#), [plot.flextable](#), [print.flextable](#), [save\\_as\\_html](#), [save\\_as\\_image](#)

**Examples**

```
ft <- flextable(head(iris, n = 2))
format(ft, type = "html")
```



---

headers\_flextable\_at\_bkm

*add flextable at a bookmark location in document's header*


---

**Description**

replace in the header of a document a paragraph containing a bookmark by a flextable. A bookmark will be considered as valid if enclosing words within a paragraph; i.e., a bookmark along two or more paragraphs is invalid, a bookmark set on a whole paragraph is also invalid, but bookmarking few words inside a paragraph is valid.

**Usage**

```
headers_flextable_at_bkm(x, bookmark, value)
```

**Arguments**

x	an rdocx object
bookmark	bookmark id
value	a flextable object

---

height

*Set flextable rows height*


---

**Description**

control rows height for a part of the flextable.

**Usage**

```
height(x, i = NULL, height, part = "body")
```

```
height_all(x, height, part = "all")
```

**Arguments**

x	flextable object
i	rows selection
height	height in inches
part	partname of the table

**height\_all**

height\_all is a convenient function for setting the same height to all rows (selected with argument part).

**See Also**

Other flextable dimensions: [autofit](#), [dim.flextable](#), [dim\\_pretty](#), [fit\\_to\\_width](#), [flextable\\_dim](#), [width](#)

**Examples**

```
ft <- flextable(iris)
ft <- height(ft, height = .3)
```

```
ft <- flextable(iris)
ft <- height_all(ft, height = .3)
```

---

hline	<i>set horizontal borders</i>
-------	-------------------------------

---

**Description**

The function is applying an horizontal border to inner content of one or all parts of a flextable. The lines are the bottom borders of selected cells.

**Usage**

```
hline(x, i = NULL, j = NULL, border = NULL, part = "body")
```

**Arguments**

x	a flextable object
i	rows selection
j	columns selection
border	border defined by a call to <a href="#">fp_border</a>
part	partname of the table (one of 'all', 'body', 'header', 'footer')

**See Also**

Other borders management: [border\\_inner\\_h](#), [border\\_inner\\_v](#), [border\\_inner](#), [border\\_outer](#), [border\\_remove](#), [border](#), [hline\\_bottom](#), [hline\\_top](#), [vline\\_left](#), [vline\\_right](#), [vline](#)

**Examples**

```
library(officer)
std_border = fp_border(color="gray")

ft <- flextable(head(iris))
ft <- border_remove(x = ft)

# add horizontal borders
ft <- hline(ft, part="all", border = std_border )
ft
```

---

hline_bottom	<i>set bottom horizontal border</i>
--------------	-------------------------------------

---

**Description**

The function is applying an horizontal border to the bottom of one or all parts of a flextable. The line is the bottom border of selected parts.

**Usage**

```
hline_bottom(x, j = NULL, border = NULL, part = "body")
```

**Arguments**

x	a flextable object
j	columns selection
border	border defined by a call to <a href="#">fp_border</a>
part	partname of the table (one of 'all', 'body', 'header', 'footer')

**See Also**

Other borders management: [border\\_inner\\_h](#), [border\\_inner\\_v](#), [border\\_inner](#), [border\\_outer](#), [border\\_remove](#), [border](#), [hline\\_top](#), [hline](#), [vline\\_left](#), [vline\\_right](#), [vline](#)

**Examples**

```
library(officer)
big_border = fp_border(color="orange", width = 3)

ft <- flextable(head(iris))
ft <- border_remove(x = ft)

# add/replace horizontal border on bottom
ft <- hline_bottom(ft, part="body", border = big_border )
ft
```

---

hline_top	<i>set top horizontal border</i>
-----------	----------------------------------

---

### Description

The function is applying an horizontal border to the top of one or all parts of a flextable. The line is the top border of selected parts.

### Usage

```
hline_top(x, j = NULL, border = NULL, part = "body")
```

### Arguments

x	a flextable object
j	columns selection
border	border defined by a call to <a href="#">fp_border</a>
part	partname of the table (one of 'all', 'body', 'header', 'footer')

### See Also

Other borders management: [border\\_inner\\_h](#), [border\\_inner\\_v](#), [border\\_inner](#), [border\\_outer](#), [border\\_remove](#), [border](#), [hline\\_bottom](#), [hline](#), [vline\\_left](#), [vline\\_right](#), [vline](#)

### Examples

```
library(officer)
big_border = fp_border(color="orange", width = 3)

ft <- flextable(head(iris))
ft <- border_remove(x = ft)

# add horizontal border on top
ft <- hline_top(ft, part="all", border = big_border )
ft
```

---

htmltools_value	<i>flextable as a div object</i>
-----------------	----------------------------------

---

### Description

get a [div](#) from a flextable object. This can be used in a shiny application.

### Usage

```
htmltools_value(x, class = "tabwid")
```

**Arguments**

x	a flextable object
class	css classes (default to "tabwid"), accepted values are "tabwid", "tabwid tabwid_left", "tabwid tabwid_right".

**See Also**

Other flextable print function: [as\\_raster](#), [docx\\_value](#), [format.flextable](#), [knit\\_print.flextable](#), [plot.flextable](#), [print.flextable](#), [save\\_as\\_html](#), [save\\_as\\_image](#)

**Examples**

```
htmltools_value(flextable(iris[1:5,]))
```

---

hyperlink_text	<i>chunk of text with hyperlink wrapper</i>
----------------	---

---

**Description**

The function lets add hyperlinks within flextable objects with function [compose](#). It should be used inside a call to [as\\_paragraph](#).

**Usage**

```
hyperlink_text(x, props = NULL, formater = format_fun, url, ...)
```

**Arguments**

x	text or any element that can be formatted as text with function provided in argument formater.
props	an <a href="#">fp_text</a> object to be used to format the text. If not specified, it will be the default value corresponding to the cell.
formater	a function that will format x as a character vector.
url	url to be used
...	additional arguments for formater function.

**See Also**

[display](#)

Other chunk elements for paragraph: [as\\_bracket](#), [as\\_b](#), [as\\_chunk](#), [as\\_image](#), [as\\_i](#), [as\\_sub](#), [as\\_sup](#), [linerange](#), [lollipop](#), [minibar](#)

**Examples**

```

dat <- data.frame(
  col = "Google it",
  href = "https://www.google.fr/search?source=hp&q=flextable+R+package",
  stringsAsFactors = FALSE)

ft <- flextable(dat)
ft <- compose( x = ft, j = "col",
  value = as_paragraph(
    "This is a link: ",
    hyperlink_text(x = col, url = href ) ) )
ft

```

---

*italic*


---

*Set italic font*


---

**Description**

change font decoration of selected rows and columns of a flextable.

**Usage**

```
italic(x, i = NULL, j = NULL, italic = TRUE, part = "body")
```

**Arguments**

<code>x</code>	a flextable object
<code>i</code>	rows selection
<code>j</code>	columns selection
<code>italic</code>	boolean value
<code>part</code>	partname of the table (one of 'all', 'body', 'header', 'footer')

**See Also**

Other sugar functions for table style: [align](#), [bg](#), [bold](#), [color](#), [empty\\_blanks](#), [fontsize](#), [font](#), [padding](#), [rotate](#), [valign](#)

**Examples**

```

ft <- flextable(mtcars)
ft <- italic(ft, italic = TRUE, part = "header")

```

---

knit\_print.flextable *Render flextable in rmarkdown*

---

### Description

Function used to render flextable in knitr/rmarkdown documents. HTML, Word and PowerPoint outputs are supported.

Function `htmltools_value` return an HTML version of the flextable, this function is to be used within Shiny applications with `renderUI()`.

### Usage

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

### Arguments

x	a flextable object
...	further arguments, not used.

### HTML chunk options

Result can be aligned with chunk option `ft.align` that accepts values 'left', 'center' and 'right'.

### Word chunk options

Result can be aligned with chunk option `ft.align` that accepts values 'left', 'center' and 'right'.

Word option 'Allow row to break across pages' can be activated with chunk option `ft.split` set to TRUE.

To specify a Word style for table caption use chunk option `tab.cap.style`. The default value is "Table Caption".

### PowerPoint chunk options

Position should be defined with options `ft.left` and `ft.top`. These are the top left coordinates of the placeholder that will contain the table. They default to `{r ft.left=1, ft.left=2}`.

### PDF chunk options

Using flextable with template 'pdf\_document' is OK if the flextable fits on one single page. The PDF output is not a real latex output but a PNG image generated with package 'webshot' or package 'webshot2'. Package 'webshot2' should be preferred as 'webshot' can have issues with some properties (i.e. bold are not rendered for some users).

To specify usage of 'webshot2', use chunk option 'webshot="webshot2"'.

**Note**

For Word (docx) output, if pandoc version  $\geq 2.0$  is used, a raw XML block with the table code will be inserted. If pandoc version  $< 2.0$  is used, an error will be raised. Insertion of images is not supported with rmarkdown for Word documents (use the package officedown instead). For PowerPoint (pptx) output, if pandoc version  $< 2.4$  is used, an error will be raised.

**Author(s)**

Maxim Nazarov

**See Also**

Other flextable print function: [as\\_raster](#), [docx\\_value](#), [format.flextable](#), [htmltools\\_value](#), [plot.flextable](#), [print.flextable](#), [save\\_as\\_html](#), [save\\_as\\_image](#)

---

linerange

*mini linerange chunk wrapper*

---

**Description**

This function is used to insert lineranges into flextable with function [compose](#). It should be used inside a call to [as\\_paragraph](#)

**Usage**

```
linerange(value, min = NULL, max = NULL, rangecol = "#CCCCCC",
  stickcol = "#FF0000", bg = "transparent", width = 1,
  height = 0.2, raster_width = 30)
```

**Arguments**

value	values containing the bar size
min	min bar size. Default min of value
max	max bar size. Default max of value
rangecol	bar color
stickcol	jauge color
bg	background color
width, height	size of the resulting png file in inches
raster_width	number of pixels used as width when interpolating value.

**Note**

PowerPoint cannot mix images and text in a paragraph, images are removed when outputting to PowerPoint format.



**See Also**

[compose](#), [as\\_paragraph](#)

Other chunk elements for paragraph: [as\\_bracket](#), [as\\_b](#), [as\\_chunk](#), [as\\_image](#), [as\\_i](#), [as\\_sub](#), [as\\_sup](#), [hyperlink\\_text](#), [lollipop](#), [minibar](#)

**Examples**

```
myft <- flextable( head(iris, n = 10 ))

myft <- compose( myft, j = 1,
  value = as_paragraph(
    linerange(value = Sepal.Length)
  ),
  part = "body")

autofit(myft)
```

---

lollipop

*mini lollipop chart chunk wrapper*


---

**Description**

This function is used to insert lollipop charts into flextable with function [compose](#). It should be used inside a call to [as\\_paragraph](#)

**Usage**

```
lollipop(value, min = NULL, max = NULL, rangecol = "#CCCCCC",
  bg = "transparent", width = 1, height = 0.2, raster_width = 30,
  positivecol = "#00CC00", negativecol = "#CC0000",
  neutralcol = "#CCCCCC", neutralrange = c(0, 0), rectanglesize = 2)
```

**Arguments**

value	values containing the bar size
min	min bar size. Default min of value
max	max bar size. Default max of value
rangecol	bar color
bg	background color
width, height	size of the resulting png file in inches
raster_width	number of pixels used as width
positivecol	box color of positive values
negativecol	box color of negative values
neutralcol	box color of neutral values
neutralrange	minimal and maximal range of neutral values (default: 0)
rectanglesize	size of the rectangle (default: 2, max: 5) when interpolating value.

**Note**

PowerPoint cannot mix images and text in a paragraph, images are removed when outputting to PowerPoint format.

**See Also**

[compose](#), [as\\_paragraph](#)

Other chunk elements for paragraph: [as\\_bracket](#), [as\\_b](#), [as\\_chunk](#), [as\\_image](#), [as\\_i](#), [as\\_sub](#), [as\\_sup](#), [hyperlink\\_text](#), [linerange](#), [minibar](#)

**Examples**

```
iris$Sepal.Ratio <- (iris$Sepal.Length - mean(iris$Sepal.Length))/mean(iris$Sepal.Length)
myft <- flextable( tail(iris, n = 10 ))

myft <- compose( myft, j = "Sepal.Ratio", value = as_paragraph(
  lollipop(value = Sepal.Ratio, min=-.25, max=.25)
),
part = "body")

autofit(myft)
```

---

merge\_at

---

*Merge flextable cells into a single one*


---

**Description**

Merge flextable cells into a single one. All rows and columns must be consecutive.

**Usage**

```
merge_at(x, i = NULL, j = NULL, part = "body")
```

**Arguments**

x	flextable object
i, j	columns and rows to merge
part	partname of the table where merge has to be done.

**See Also**

Other flextable merging function: [merge\\_h\\_range](#), [merge\\_h](#), [merge\\_none](#), [merge\\_v](#)

**Examples**

```
ft_merge <- flextable( head( mtcars ), cwidth = .5 )
ft_merge <- merge_at( ft_merge, i = 1:2, j = 1:2 )
ft_merge
```

---

merge_h	<i>Merge flextable cells horizontally</i>
---------	---

---

**Description**

Merge flextable cells horizontally when consecutive cells have identical values. Text of formatted values are used to compare values.

**Usage**

```
merge_h(x, i = NULL, part = "body")
```

**Arguments**

x	flextable object
i	rows where cells have to be merged.
part	partname of the table where merge has to be done.

**See Also**

Other flextable merging function: [merge\\_at](#), [merge\\_h\\_range](#), [merge\\_none](#), [merge\\_v](#)

**Examples**

```
dummy_df <- data.frame( col1 = letters,
  col2 = letters, stringsAsFactors = FALSE )
ft_merge <- flextable(dummy_df)
ft_merge <- merge_h(x = ft_merge)
ft_merge
```

---

merge_h_range	<i>rowwise merge of a range of columns</i>
---------------	--

---

**Description**

Merge flextable columns into a single one for each selected rows. All columns must be consecutive.

**Usage**

```
merge_h_range(x, i = NULL, j1 = NULL, j2 = NULL, part = "body")
```

**Arguments**

x	flextable object
i	selected rows
j1, j2	selected columns that will define the range of columns to merge.
part	partname of the table where merge has to be done.

**See Also**

Other flextable merging function: [merge\\_at](#), [merge\\_h](#), [merge\\_none](#), [merge\\_v](#)

**Examples**

```
ft <- flextable( head( mtcars ), cwidth = .5 )
ft <- merge_h_range( ft, i = ~ cyl == 6, j1 = "am", j2 = "carb" )
ft
```

---

merge\_none

*Delete flextable merging informations*

---

**Description**

Delete all merging informations from a flextable.

**Usage**

```
merge_none(x, part = "all")
```

**Arguments**

x                    flextable object  
part                  partname of the table where merge has to be done.

**See Also**

Other flextable merging function: [merge\\_at](#), [merge\\_h\\_range](#), [merge\\_h](#), [merge\\_v](#)

**Examples**

```
typology <- data.frame(
  col_keys = c( "Sepal.Length", "Sepal.Width", "Petal.Length", "Petal.Width", "Species" ),
  what = c("Sepal", "Sepal", "Petal", "Petal", "Species"),
  measure = c("Length", "Width", "Length", "Width", "Species"),
  stringsAsFactors = FALSE )

ft <- flextable( head( iris ) )
ft <- set_header_df(ft, mapping = typology, key = "col_keys" )
ft <- merge_v(ft, j = c("Species"))

ft <- theme_tron_legacy( merge_none( ft ) )
ft
```

---

merge_v	<i>Merge flextable cells vertically</i>
---------	---

---

### Description

Merge flextable cells vertically when consecutive cells have identical values. Text of formatted values are used to compare values.

### Usage

```
merge_v(x, j = NULL, target = NULL, part = "body")
```

### Arguments

x	flextable object
j	column to used to find consecutive values to be merged.
target	columns names where cells have to be merged.
part	partname of the table where merge has to be done.

### See Also

Other flextable merging function: [merge\\_at](#), [merge\\_h\\_range](#), [merge\\_h](#), [merge\\_none](#)

### Examples

```
ft_merge <- flextable(mtcars)
ft_merge <- merge_v(ft_merge, j = c("gear", "carb"))
ft_merge

data_ex <- structure(list(srdr_id = c(
  "175124", "175124", "172525", "172525",
  "172545", "172545", "172609", "172609", "172609"
), substances = c(
  "alcohol",
  "alcohol", "alcohol", "alcohol", "cannabis",
  "cannabis", "alcohol\n cannabis\n other drugs",
  "alcohol\n cannabis\n other drugs",
  "alcohol\n cannabis\n other drugs"
), full_name = c(
  "TAU", "MI", "TAU", "MI (parent)", "TAU", "MI",
  "TAU", "MI", "MI"
), article_arm_name = c(
  "Control", "WISEteens",
  "Treatment as usual", "Brief MI (b-MI)", "Assessed control",
  "Intervention", "Control", "Computer BI", "Therapist BI"
)), row.names = c(
  NA,
  -9L
```

```

), class = c("tbl_df", "tbl", "data.frame"))
ft <- flextable(data_ex)
ft <- theme_box(ft)
merge_v(ft, j = "srdr_id",
  target = c("srdr_id", "substances"))

```

---

minibar

*mini barplots chunk wrapper*


---

## Description

This function is used to insert bars into flextable with function [compose](#). It should be used inside a call to [as\\_paragraph](#)

## Usage

```

minibar(value, max = NULL, barcol = "#CCCCCC", bg = "transparent",
  width = 1, height = 0.2)

```

## Arguments

value	values containing the bar size
max	max bar size
barcol	bar color
bg	background color
width, height	size of the resulting png file in inches

## Note

PowerPoint cannot mix images and text in a paragraph, images are removed when outputting to PowerPoint format.

## See Also

[compose](#), [as\\_paragraph](#)

Other chunk elements for paragraph: [as\\_bracket](#), [as\\_b](#), [as\\_chunk](#), [as\\_image](#), [as\\_i](#), [as\\_sub](#), [as\\_sup](#), [hyperlink\\_text](#), [linerange](#), [lollipop](#)

## Examples

```

myft <- flextable( head(iris, n = 10 ))

myft <- compose( myft, j = 1,
  value = as_paragraph(
    minibar(value = Sepal.Length, max = max(Sepal.Length))
  ),
  part = "body")

autofit(myft)

```

---

padding	<i>Set paragraph paddings</i>
---------	-------------------------------

---

### Description

change paddings of selected rows and columns of a flextable.

### Usage

```
padding(x, i = NULL, j = NULL, padding = NULL, padding.top = NULL,  
padding.bottom = NULL, padding.left = NULL, padding.right = NULL,  
part = "body")
```

### Arguments

x	a flextable object
i	rows selection
j	columns selection
padding	padding (shortcut for top, bottom, left and right)
padding.top	padding top
padding.bottom	padding bottom
padding.left	padding left
padding.right	padding right
part	partname of the table (one of 'all', 'body', 'header', 'footer')

### See Also

Other sugar functions for table style: [align](#), [bg](#), [bold](#), [color](#), [empty\\_blanks](#), [fontsize](#), [font](#), [italic](#), [rotate](#), [valign](#)

### Examples

```
ft <- flextable(mtcars)  
ft <- padding(ft, padding.top = 4)
```

---

ph\_with.flextable      *add a flextable into a PowerPoint slide*

---

## Description

Add a flextable in a PowerPoint document object produced by [read\\_pptx](#).

## Usage

```
## S3 method for class 'flextable'  
ph_with(x, value, location, ...)
```

## Arguments

x	a pptx device
value	flextable object
location	a location for a placeholder. See <a href="#">ph_location_type</a> for example.
...	unused arguments.

## Note

The width and height of the table can not be set with location. Use functions [width](#), [height](#), [autofit](#) and [dim\\_pretty](#) instead. The overall size is resulting from cells, paragraphs and text properties (i.e. padding, font size, border widths).

## Examples

```
library(officer)  
  
ft = flextable(head(iris))  
  
doc <- read_pptx()  
doc <- add_slide(doc, "Title and Content", "Office Theme")  
doc <- ph_with(doc, ft, location = ph_location_left())  
  
fileout <- tempfile(fileext = ".pptx")  
print(doc, target = fileout)
```



---

ph\_with\_flextable      *add flextable into a PowerPoint slide*

---

### Description

add a flextable as a new shape in the current slide.

These functions will be deprecated in the next release and function `ph_with.flextable` should be used instead.

### Usage

```
ph_with_flextable(x, value, type = "body", index = 1)
```

```
ph_with_flextable_at(x, value, left, top)
```

### Arguments

x	an rpptx device
value	flextable object
type	placeholder type
index	placeholder index (integer). This is to be used when a placeholder type is not unique in the current slide, e.g. two placeholders with type 'body'.
left, top	location of flextable on the slide in inches

### Note

The width and height of the table can not be set with this function. Use functions `width`, `height`, `autofit` and `dim_pretty` instead. The overall size is resulting from cells, paragraphs and text properties (i.e. padding, font size, border widths).

---

plot.flextable      *plot a flextable*

---

### Description

save a flextable as an image and display the result in a new R graphics window.

### Usage

```
## S3 method for class 'flextable'
plot(x, zoom = 2, expand = 2, ...)
```

**Arguments**

x	a flextable object
zoom, expand	parameters used by webshot function.
...	additional parameters sent to plot function

**Note**

This function requires packages: webshot and magick.

**See Also**

Other flextable print function: [as\\_raster](#), [docx\\_value](#), [format.flextable](#), [htmltools\\_value](#), [knit\\_print.flextable](#), [print.flextable](#), [save\\_as\\_html](#), [save\\_as\\_image](#)

**Examples**

```
ft <- flextable( head( mtcars ) )
ft <- autofit(ft)
## Not run:
if( require("webshot") ){
  plot(ft)
}

## End(Not run)
```

---

print.flextable      *flextable printing*

---

**Description**

print a flextable object to format html, docx, pptx or as text (not for display but for informative purpose). This function is to be used in an interactive context.

**Usage**

```
## S3 method for class 'flextable'
print(x, preview = "html", ...)
```

**Arguments**

x	flextable object
preview	preview type, one of c("html", "pptx", "docx", "log"). When "log" is used, a description of the flextable is printed.
...	unused argument

**Note**

When argument `preview` is set to "docx" or "pptx", an external client linked to these formats (Office is installed) is used to edit a document. The document is saved in the temporary directory of the R session and will be removed when R session will be ended.

When argument `preview` is set to "html", an external client linked to these HTML format is used to display the table. If RStudio is used, the Viewer is used to display the table.

Note also that a print method is used when flextable are used within R markdown documents. See [knit\\_print.flextable](#).

**See Also**

Other flextable print function: [as\\_raster](#), [docx\\_value](#), [format.flextable](#), [htmltools\\_value](#), [knit\\_print.flextable](#), [plot.flextable](#), [save\\_as\\_html](#), [save\\_as\\_image](#)

---

proc_freq	<i>frequency table as flextable</i>
-----------	-------------------------------------

---

**Description**

This function compute a two way contingency table and make a flextable with the result.

**Usage**

```
proc_freq(x, row, col, main = "", include.row_percent = TRUE,
  include.column_percent = TRUE, include.table_percent = TRUE,
  include.column_total = TRUE, include.row_total = TRUE,
  include.header_row = TRUE, weight = NULL)
```

**Arguments**

<code>x</code>	data.frame object
<code>row</code>	character column names for row
<code>col</code>	character column names for column
<code>main</code>	character title
<code>include.row_percent</code>	boolean whether to include the row percents; defaults to TRUE
<code>include.column_percent</code>	boolean whether to include the column percents; defaults to TRUE
<code>include.table_percent</code>	boolean whether to include the table percents; defaults to TRUE
<code>include.column_total</code>	boolean whether to include the row of column totals; defaults to TRUE
<code>include.row_total</code>	boolean whether to include the column of row totals; defaults to TRUE
<code>include.header_row</code>	boolean whether to include the header row; defaults to TRUE
<code>weight</code>	character column name for weight

**Author(s)**

Titouan Robert

**Examples**

```
proc_freq(mtcars, "vs", "gear")
proc_freq(mtcars, "gear", "vs")
proc_freq(mtcars, "gear", "vs", weight = "wt")
proc_freq(mtcars, "gear", "vs", "My title")
```

---

rotate	<i>rotate cell text</i>
--------	-------------------------

---

**Description**

apply a rotation to cell text

**Usage**

```
rotate(x, i = NULL, j = NULL, rotation, align = "center",
part = "body")
```

**Arguments**

x	a flextable object
i	rows selection
j	columns selection
rotation	one of "lrb", "tbl", "btlr"
align	vertical alignment of paragraph within cell, one of "center" or "top" or "bottom".
part	partname of the table (one of 'all', 'body', 'header', 'footer')

**Details**

One common case is to rotate text to minimise column space. When rotating, paragraph alignments will remain the same and often right aligned ( with an effect of top aligned when rotated). Use `align(..., align = "center")` to center rotated text.

When function `autofit` is used, the rotation will be ignored.

**See Also**

Other sugar functions for table style: [align](#), [bg](#), [bold](#), [color](#), [empty\\_blanks](#), [fontsize](#), [font](#), [italic](#), [padding](#), [valign](#)

### Examples

```
ft <- flextable(head(iris))
ft <- rotate(ft, rotation = "tbr1", part = "header", align = "center")
ft <- align(ft, align = "center")
ft <- autofit(ft)
ft <- height(ft, height = max(dim_pretty(ft, part = "header")$widths), part = "header")
```

---

save_as_html	<i>save a flextable in an HTML file</i>
--------------	---

---

### Description

save a flextable in an HTML file. This function has been implemented to help users that do not understand R Markdown. It is highly recommended to use R Markdown instead.

### Usage

```
save_as_html(x, path)
```

### Arguments

x	a flextable object
path	HTML file to be created

### See Also

Other flextable print function: [as\\_raster](#), [docx\\_value](#), [format.flextable](#), [htmltools\\_value](#), [knit\\_print.flextable](#), [plot.flextable](#), [print.flextable](#), [save\\_as\\_image](#)

### Examples

```
ft <- flextable( head( mtcars ) )
tf <- tempfile(fileext = ".html")
save_as_html(ft, tf)
```

---

save_as_image	<i>save a flextable as an image</i>
---------------	-------------------------------------

---

### Description

save a flextable as a png, pdf or jpeg image.

Image generated with package 'webshot' or package 'webshot2'. Package 'webshot2' should be preferred as 'webshot' can have issues with some properties (i.e. bold are not rendered for some users).

**Usage**

```
save_as_image(x, path, zoom = 3, expand = 10, webshot = "webshot")
```

**Arguments**

x	a flextable object
path	image file to be created. It should end with .png, .pdf, or .jpeg.
zoom, expand	parameters used by webshot function.
webshot	webshot package as a scalar character, one of "webshot" or "webshot2".

**Note**

This function requires package webshot.

**See Also**

Other flextable print function: [as\\_raster](#), [docx\\_value](#), [format.flextable](#), [htmltools\\_value](#), [knit\\_print.flextable](#), [plot.flextable](#), [print.flextable](#), [save\\_as\\_html](#)

**Examples**

```
ft <- flextable( head( mtcars ) )
ft <- autofit(ft)
tf <- tempfile(fileext = ".png")
## Not run:
if( require("webshot") ){
  save_as_image(x = ft, path = "myimage.png")
}

## End(Not run)
```

---

set\_caption

*set caption*

---

**Description**

set caption value in flextable

**Usage**

```
set_caption(x, caption)
```

**Arguments**

x	flextable object
caption	caption value

**Note**

this will have an effect only when output is HTML or Word document.

**Examples**

```
ft <- flextable( head( iris ) )
ft <- set_caption(ft, "my caption")
ft
```

---

set_formatter	<i>set column formatter functions</i>
---------------	---------------------------------------

---

**Description**

Define formatter functions associated to each column key. Functions have a single argument (the vector) and are returning the formatted values as a character vector.

**Usage**

```
set_formatter(x, ..., values = NULL, part = "body")

set_formatter_type(x, fmt_double = "%.03f", fmt_integer = "%.0f",
  fmt_date = "%Y-%m-%d", fmt_datetime = "%Y-%m-%d %H:%M:%S",
  true = "true", false = "false", na_str = "")
```

**Arguments**

x	a flextable object
...	Name-value pairs of functions, names should be existing col_key values
values	a list of name-value pairs of functions, names should be existing col_key values. If values is supplied argument ... is ignored.
part	partname of the table (one of 'body' or 'header' or 'footer')
fmt_double, fmt_integer	arguments used by sprintf to format double and integer columns.
fmt_date, fmt_datetime	arguments used by format to format date and date time columns.
false, true	string to be used for logical columns
na_str	string for NA values

**set\_formatter\_type**

set\_formatter\_type is an helper function to quickly define formatter functions regarding to column types.

**Examples**

```
ft <- flextable( head( iris ) )
ft <- set_formatter( x = ft,
  Sepal.Length = function(x) sprintf("%.02f", x),
  Sepal.Width = function(x) sprintf("%.04f", x)
)
ft <- theme_vanilla( ft )
ft
```

---

set\_header\_footer\_df *Set flextable's header or footer rows*

---

**Description**

Use a data.frame to specify flextable's header or footer rows.

The data.frame must contain a column whose values match flextable col\_keys argument, this column will be used as join key. The other columns will be displayed as header or footer rows. The leftmost column is used as the top header/footer row and the rightmost column is used as the bottom header/footer row.

**Usage**

```
set_header_df(x, mapping = NULL, key = "col_keys")
```

```
set_footer_df(x, mapping = NULL, key = "col_keys")
```

**Arguments**

x	a flextable object
mapping	a data.frame specifying for each colname content of the column.
key	column to use as key when joining data_mapping.

**See Also**

Other headers and footers: [add\\_header\\_lines](#), [add\\_header\\_row](#), [add\\_header](#)

**Examples**

```
typology <- data.frame(
  col_keys = c( "Sepal.Length", "Sepal.Width", "Petal.Length",
    "Petal.Width", "Species" ),
  what = c("Sepal", "Sepal", "Petal", "Petal", "Species"),
  measure = c("Length", "Width", "Length", "Width", "Species"),
  stringsAsFactors = FALSE )

ft <- flextable( head( iris ) )
ft <- set_header_df(ft, mapping = typology, key = "col_keys" )
```



```

ft <- merge_h(ft, part = "header")
ft <- merge_v(ft, j = "Species", part = "header")
ft <- theme_vanilla(ft)

typology <- data.frame(
  col_keys = c( "Sepal.Length", "Sepal.Width", "Petal.Length",
               "Petal.Width", "Species" ),
  unit = c("cm", "cm", "cm", "cm", "" ),
  stringsAsFactors = FALSE )
ft <- set_footer_df(ft, mapping = typology, key = "col_keys" )
ft <- italic(ft, italic = TRUE, part = "footer" )
ft <- theme_booktabs(ft)
ft

```

---

set\_header\_labels      *Set flextable's headers labels*

---

### Description

This function set labels for specified columns in a single row header of a flextable.

### Usage

```
set_header_labels(x, ..., values = NULL)
```

### Arguments

x	a flextable object
...	named arguments (names are data colnames), each element is a single character value specifying label to use.
values	a named list (names are data colnames), each element is a single character value specifying label to use. If provided, argument ... will be ignored.

### Examples

```

ft_1 <- flextable( head( iris ))
ft_1 <- set_header_labels(ft_1, Sepal.Length = "Sepal length",
  Sepal.Width = "Sepal width", Petal.Length = "Petal length",
  Petal.Width = "Petal width"
)

ft_2 <- flextable( head( iris ))
ft_2 <- set_header_labels(ft_2,
  values = list(Sepal.Length = "Sepal length",
               Sepal.Width = "Sepal width",
               Petal.Length = "Petal length",
               Petal.Width = "Petal width" ) )
ft_2

```

---

style	<i>Set flextable style</i>
-------	----------------------------

---

### Description

Modify flextable text, paragraphs and cells formatting properties. It allows to specify a set of formatting properties for a selection instead of using multiple functions (.i.e bold, italic, bg) that should all be applied to the same selection of rows and columns.

### Usage

```
style(x, i = NULL, j = NULL, pr_t = NULL, pr_p = NULL,
      pr_c = NULL, part = "body")
```

### Arguments

x	a flextable object
i	rows selection
j	columns selection
pr_t	object(s) of class fp_text
pr_p	object(s) of class fp_par
pr_c	object(s) of class fp_cell
part	partname of the table (one of 'all', 'body', 'header' or 'footer')

### Examples

```
library(officer)
def_cell <- fp_cell(border = fp_border(color="#00FFFF"))

def_par <- fp_par(text.align = "center")

ft <- flextable(mtcars)

ft <- style( ft, pr_c = def_cell, pr_p = def_par, part = "all")
ft <- style(ft, ~ drat > 3.5, ~ vs + am + gear + carb,
           pr_t = fp_text(color="red", italic = TRUE) )

ft
```

---

theme_alafoli	<i>Apply alafoli theme</i>
---------------	----------------------------

---

**Description**

Apply theme alafoli to a flextable

**Usage**

```
theme_alafoli(x)
```

**Arguments**

x                    a flextable object

**See Also**

Other flextable theme: [theme\\_booktabs](#), [theme\\_box](#), [theme\\_tron\\_legacy](#), [theme\\_tron](#), [theme\\_vader](#), [theme\\_vanilla](#), [theme\\_zebra](#)

**Examples**

```
ft <- flextable(iris)
ft <- theme_alafoli(ft)
```

---

theme_booktabs	<i>Apply booktabs theme</i>
----------------	-----------------------------

---

**Description**

Apply theme booktabs to a flextable

**Usage**

```
theme_booktabs(x, fontsize = 11)
```

**Arguments**

x                    a flextable object  
fontsize            font size in pixel

**See Also**

Other flextable theme: [theme\\_alafoli](#), [theme\\_box](#), [theme\\_tron\\_legacy](#), [theme\\_tron](#), [theme\\_vader](#), [theme\\_vanilla](#), [theme\\_zebra](#)

**Examples**

```
ft <- flextable(iris)
ft <- theme_booktabs(ft)
```

---

theme_box	<i>Apply box theme</i>
-----------	------------------------

---

**Description**

Apply theme box to a flextable

**Usage**

```
theme_box(x)
```

**Arguments**

x a flextable object

**See Also**

Other flextable theme: [theme\\_alafoli](#), [theme\\_booktabs](#), [theme\\_tron\\_legacy](#), [theme\\_tron](#), [theme\\_vader](#), [theme\\_vanilla](#), [theme\\_zebra](#)

**Examples**

```
ft <- flextable(iris)
ft <- theme_box(ft)
```

---

theme_tron	<i>Apply tron theme</i>
------------	-------------------------

---

**Description**

Apply theme tron to a flextable

**Usage**

```
theme_tron(x)
```

**Arguments**

x a flextable object

**See Also**

Other flextable theme: [theme\\_alafoli](#), [theme\\_booktabs](#), [theme\\_box](#), [theme\\_tron\\_legacy](#), [theme\\_vader](#), [theme\\_vanilla](#), [theme\\_zebra](#)

**Examples**

```
ft <- flextable(iris)
ft <- theme_tron(ft)
```

---

theme_tron_legacy	<i>Apply tron legacy theme</i>
-------------------	--------------------------------

---

**Description**

Apply theme tron legacy to a flextable

**Usage**

```
theme_tron_legacy(x)
```

**Arguments**

x a flextable object

**See Also**

Other flextable theme: [theme\\_alafoli](#), [theme\\_booktabs](#), [theme\\_box](#), [theme\\_tron](#), [theme\\_vader](#), [theme\\_vanilla](#), [theme\\_zebra](#)

**Examples**

```
ft <- flextable(iris)
ft <- theme_tron_legacy(ft)
```

---

theme_vader	<i>Apply Sith Lord Darth Vader</i>
-------------	------------------------------------

---

**Description**

Apply Sith Lord Darth Vader theme to a flextable

**Usage**

```
theme_vader(x, fontsize = 11)
```

**Arguments**

x                    a flextable object  
fontsize            font size in pixel

**See Also**

Other flextable theme: [theme\\_alafoli](#), [theme\\_booktabs](#), [theme\\_box](#), [theme\\_tron\\_legacy](#), [theme\\_tron](#), [theme\\_vanilla](#), [theme\\_zebra](#)

**Examples**

```
ft <- flextable(iris)
ft <- theme_vader(ft)
```

---

theme_vanilla	<i>Apply vanilla theme</i>
---------------	----------------------------

---

**Description**

Apply theme vanilla to a flextable

**Usage**

```
theme_vanilla(x)
```

**Arguments**

x                    a flextable object

**See Also**

Other flextable theme: [theme\\_alafoli](#), [theme\\_booktabs](#), [theme\\_box](#), [theme\\_tron\\_legacy](#), [theme\\_tron](#), [theme\\_vader](#), [theme\\_zebra](#)

**Examples**

```
ft <- flextable(iris)
ft <- theme_vanilla(ft)
```

---

theme_zebra	<i>Apply zebra theme</i>
-------------	--------------------------

---

**Description**

Apply theme zebra to a flextable

**Usage**

```
theme_zebra(x, odd_header = "#CFCFCF", odd_body = "#EFEFEF",
  even_header = "transparent", even_body = "transparent")
```

**Arguments**

x                    a flextable object  
 odd\_header, odd\_body, even\_header, even\_body  
                     odd/even colors for table header and body

**See Also**

Other flextable theme: [theme\\_alafoli](#), [theme\\_booktabs](#), [theme\\_box](#), [theme\\_tron\\_legacy](#), [theme\\_tron](#), [theme\\_vader](#), [theme\\_vanilla](#)

**Examples**

```
ft <- flextable(iris)
ft <- theme_zebra(ft)
```

---

valign	<i>Set vertical alignment</i>
--------	-------------------------------

---

**Description**

change vertical alignment of selected rows and columns of a flextable.

**Usage**

```
valign(x, i = NULL, j = NULL, valign = "center", part = "body")
```

**Arguments**

x                    a flextable object  
 i                    rows selection  
 j                    columns selection  
 valign              vertical alignment of paragraph within cell, one of "center" or "top" or "bottom".  
 part                partname of the table (one of 'all', 'body', 'header', 'footer')

**See Also**

Other sugar functions for table style: [align](#), [bg](#), [bold](#), [color](#), [empty\\_blanks](#), [fontsize](#), [font](#), [italic](#), [padding](#), [rotate](#)

**Examples**

```
ft <- flextable(iris[c(1:3, 51:53, 101:103),])
ft <- theme_box(ft)
ft <- merge_v( ft, j = 5)
ft <- valign(ft, j = 5, valign = "top", part = "all")
ft
```

---

vline

*set vertical borders*


---

**Description**

The function is applying vertical borders to inner content of one or all parts of a flextable. The lines are the right borders of selected cells.

**Usage**

```
vline(x, i = NULL, j = NULL, border = NULL, part = "all")
```

**Arguments**

x	a flextable object
i	rows selection
j	columns selection
border	border defined by a call to <a href="#">fp_border</a>
part	partname of the table (one of 'all', 'body', 'header', 'footer')

**See Also**

Other borders management: [border\\_inner\\_h](#), [border\\_inner\\_v](#), [border\\_inner](#), [border\\_outer](#), [border\\_remove](#), [border](#), [hline\\_bottom](#), [hline\\_top](#), [hline](#), [vline\\_left](#), [vline\\_right](#)

**Examples**

```
library(officer)
std_border = fp_border(color="orange")

ft <- flextable(head(iris))
ft <- border_remove(x = ft)

# add vertical borders
ft <- vline(ft, border = std_border )
ft
```



---

vline_left	<i>set flextable left vertical borders</i>
------------	--

---

**Description**

The function is applying vertical borders to the left side of one or all parts of a flextable. The line is the left border of selected cells of the first column.

**Usage**

```
vline_left(x, i = NULL, border = NULL, part = "all")
```

**Arguments**

x	a flextable object
i	rows selection
border	border defined by a call to <a href="#">fp_border</a>
part	partname of the table (one of 'all', 'body', 'header', 'footer')

**See Also**

Other borders management: [border\\_inner\\_h](#), [border\\_inner\\_v](#), [border\\_inner](#), [border\\_outer](#), [border\\_remove](#), [border](#), [hline\\_bottom](#), [hline\\_top](#), [hline](#), [vline\\_right](#), [vline](#)

**Examples**

```
library(officer)
std_border = fp_border(color="orange")

ft <- flextable(head(iris))
ft <- border_remove(x = ft)

# add vertical border on the left side of the table
ft <- vline_left(ft, border = std_border )
ft
```

---

vline_right	<i>set flextable right vertical borders</i>
-------------	---

---

**Description**

The function is applying vertical borders to the right side of one or all parts of a flextable. The line is the right border of selected cells of the last column.

**Usage**

```
vline_right(x, i = NULL, border = NULL, part = "all")
```

**Arguments**

x	a flextable object
i	rows selection
border	border defined by a call to <a href="#">fp_border</a>
part	partname of the table (one of 'all', 'body', 'header', 'footer')

**See Also**

Other borders management: [border\\_inner\\_h](#), [border\\_inner\\_v](#), [border\\_inner](#), [border\\_outer](#), [border\\_remove](#), [border](#), [hline\\_bottom](#), [hline\\_top](#), [hline](#), [vline\\_left](#), [vline](#)

**Examples**

```
library(officer)
std_border = fp_border(color="orange")

ft <- flextable(head(iris))
ft <- border_remove(x = ft)

# add vertical border on the left side of the table
ft <- vline_right(ft, border = std_border )
ft
```

---

void

*Delete flextable content*


---

**Description**

Set content display as a blank " ".

**Usage**

```
void(x, j = NULL, part = "body")
```

**Arguments**

x	flextable object
j	columns selection
part	partname of the table

**Examples**

```
ft <- flextable(mtcars)
ft <- void(ft, ~ vs + am + gear + carb )
```

---

width	<i>Set flextable columns width</i>
-------	------------------------------------

---

**Description**

control columns width

**Usage**

```
width(x, j = NULL, width)
```

**Arguments**

x	flextable object
j	columns selection
width	width in inches

**Details**

Heights are not used when flextable is been rendered into HTML.

**See Also**

Other flextable dimensions: [autofit](#), [dim.flextable](#), [dim\\_pretty](#), [fit\\_to\\_width](#), [flextable\\_dim](#), [height](#)

**Examples**

```
ft <- flextable(iris)
ft <- width(ft, width = 1)
```

---

xtable_to_flextable	<i>get a flextable from a xtable object</i>
---------------------	---

---

**Description**

Get a flextable object from a xtable object.

**Usage**

```
xtable_to_flexitable(x, text.properties = officer::fp_text(),
  format.args = getOption("xtable.format.args", NULL),
  rowname_col = "rowname",
  hline.after = getOption("xtable.hline.after", c(-1, 0, nrow(x))),
  NA.string = getOption("xtable.NA.string", ""),
  include.rownames = TRUE,
  rotate.colnames = getOption("xtable.rotate.colnames", FALSE))
```

**Arguments**

x	xtable object
text.properties	default text formatting properties
format.args	List of arguments for the formatC function. See argument format.args of print.xtable. Not yet implemented.
rowname_col	colname used for row names column
hline.after	see ?print.xtable.
NA.string	see ?print.xtable.
include.rownames	see ?print.xtable.
rotate.colnames	see ?print.xtable.

**Examples**

```
library(officer)
if( require("xtable") ){

  data(tli)
  tli.table <- xtable(tli[1:10, ])
  align(tli.table) <- rep("r", 6)
  align(tli.table) <- "|r|r|clr|r|"
  ft <- xtable_to_flexitable(
    tli.table,
    rotate.colnames = TRUE,
    include.rownames = FALSE)
  ft <- height(ft, i = 1, part = "header", height = 1)
  ft

  Grade3 <- c("A", "B", "B", "A", "B", "C", "C", "D", "A", "B",
    "C", "C", "C", "D", "B", "B", "D", "C", "C", "D")
  Grade6 <- c("A", "A", "A", "B", "B", "B", "B", "B", "C", "C",
    "A", "C", "C", "C", "D", "D", "D", "D", "D")
  Cohort <- table(Grade3, Grade6)
  ft <- xtable_to_flexitable(xtable(Cohort))
  ft <- set_header_labels(ft, rowname = "Grade 3")
  ft <- autofit(ft)
```

```
ft <- add_header(ft, A = "Grade 6")
ft <- merge_at(ft, i = 1, j = seq_len( ncol(Cohort) ) + 1,
  part = "header" )
ft <- bold(ft, j = 1, bold = TRUE, part = "body")
ft <- height_all(ft, part = "header", height = .4)
ft

temp.ts <- ts(cumsum(1 + round(rnorm(100), 0)),
  start = c(1954, 7), frequency = 12)
xtable_to_flexable(x = xtable(temp.ts, digits = 0),
  NA.string = "-")
}
```

# Index

`add_footer` (`add_header`), 4  
`add_footer_lines` (`add_header_lines`), 5  
`add_footer_row` (`add_header_row`), 6  
`add_header`, 4, 6, 64  
`add_header_lines`, 5, 5, 6, 64  
`add_header_row`, 5, 6, 6, 64  
`align`, 7, 18, 20, 28, 33, 38, 46, 55, 60, 72  
`align_nottext_col` (`align`), 7  
`align_text_col` (`align`), 7  
`as_b`, 8, 9, 10, 12, 13, 16, 45, 49, 50, 54  
`as_bracket`, 8, 9, 10, 12, 13, 16, 45, 49, 50, 54  
`as_chunk`, 8, 9, 9, 12–14, 16, 45, 49, 50, 54  
`as_flextable`, 10, 12  
`as_grouped_data`, 11  
`as_i`, 8–10, 12, 13, 16, 45, 49, 50, 54  
`as_image`, 8–10, 12, 13, 14, 16, 45, 49, 50, 54  
`as_paragraph`, 8, 9, 12, 13, 14, 16, 29, 39, 45, 48–50, 54  
`as_raster`, 15, 33, 40, 45, 48, 58, 59, 61, 62  
`as_sub`, 8–10, 12, 13, 15, 16, 45, 49, 50, 54  
`as_sup`, 8–10, 12, 13, 16, 16, 45, 49, 50, 54  
`autofit`, 17, 30, 31, 34, 37, 42, 56, 57, 75  
  
`bg`, 7, 18, 20, 28, 33, 38, 46, 55, 60, 72  
`body_add_flextable`, 18  
`body_replace_flextable_at_bkm` (`body_add_flextable`), 18  
`bold`, 7, 18, 19, 28, 33, 38, 46, 55, 60, 72  
`border`, 20, 21–24, 42–44, 72–74  
`border_inner`, 20, 21, 22–24, 42–44, 72–74  
`border_inner_h`, 20, 21, 21, 22–24, 42–44, 72–74  
`border_inner_v`, 20–22, 22, 23, 24, 42–44, 72–74  
`border_outer`, 20–22, 23, 24, 42–44, 72–74  
`border_remove`, 20–23, 24, 42–44, 72–74  
  
`colformat_char`, 24, 26, 27, 29  
`colformat_int`, 25, 25, 26, 27, 29  
`colformat_lgl`, 25, 26, 26, 27, 29  
  
`colformat_num`, 25, 26, 27, 29  
`color`, 7, 18, 20, 28, 33, 38, 46, 55, 60, 72  
`compose`, 4, 9, 13, 14, 25–27, 29, 32, 45, 48–50, 54  
  
`delete_part`, 30  
`dim.flextable`, 17, 30, 31, 34, 37, 42, 75  
`dim_pretty`, 17, 30, 31, 34, 37, 42, 56, 57, 75  
`display`, 31, 45  
`div`, 44  
`docx_value`, 15, 33, 40, 45, 48, 58, 59, 61, 62  
  
`empty_blanks`, 7, 18, 20, 28, 33, 38, 46, 55, 60, 72  
  
`fit_to_width`, 17, 30, 31, 34, 37, 42, 75  
`fix_border_issues`, 35  
`flextable`, 4, 35  
`flextable-package`, 4  
`flextable_dim`, 17, 30, 31, 34, 37, 42, 75  
`font`, 7, 18, 20, 28, 33, 37, 38, 46, 55, 60, 72  
`fontsize`, 7, 18, 20, 28, 33, 38, 38, 46, 55, 60, 72  
  
`footers_flextable_at_bkm`, 39  
`footnote`, 39  
`format.flextable`, 15, 33, 40, 45, 48, 58, 59, 61, 62  
`formatC`, 25, 27  
`fp_border`, 21–23, 42–44, 72–74  
`fp_text`, 10, 32, 45  
  
`headers_flextable_at_bkm`, 41  
`height`, 17, 30, 31, 34, 37, 41, 56, 57, 75  
`height_all` (`height`), 41  
`hline`, 20–24, 42, 43, 44, 72–74  
`hline_bottom`, 20–24, 42, 43, 44, 72–74  
`hline_top`, 20–24, 42, 43, 44, 72–74  
`htmltools_value`, 15, 33, 40, 44, 48, 58, 59, 61, 62  
`hyperlink_text`, 8–10, 12–14, 16, 45, 49, 50, 54

- italic*, [7](#), [18](#), [20](#), [28](#), [33](#), [38](#), [46](#), [55](#), [60](#), [72](#)
- `knit_print.flextable`, [15](#), [33](#), [40](#), [45](#), [47](#), [58](#), [59](#), [61](#), [62](#)
- `linerrange`, [8–10](#), [12](#), [13](#), [16](#), [45](#), [48](#), [50](#), [54](#)
- `lollipop`, [8–10](#), [12](#), [13](#), [16](#), [45](#), [49](#), [49](#), [54](#)
- `merge_at`, [50](#), [51–53](#)
- `merge_h`, [4](#), [50](#), [51](#), [52](#), [53](#)
- `merge_h_range`, [50](#), [51](#), [51](#), [52](#), [53](#)
- `merge_none`, [50–52](#), [52](#), [53](#)
- `merge_v`, [4](#), [50–52](#), [53](#)
- `minibar`, [8–10](#), [12–14](#), [16](#), [45](#), [49](#), [50](#), [54](#)
- `mk_par` (`compose`), [29](#)
- 
- `padding`, [7](#), [18](#), [20](#), [28](#), [33](#), [38](#), [46](#), [55](#), [60](#), [72](#)
- `ph_location_type`, [56](#)
- `ph_with.flextable`, [56](#), [57](#)
- `ph_with_flextable`, [57](#)
- `ph_with_flextable_at`  
(`ph_with_flextable`), [57](#)
- `plot.flextable`, [15](#), [33](#), [40](#), [45](#), [48](#), [57](#), [59](#), [61](#), [62](#)
- `print.flextable`, [15](#), [33](#), [40](#), [45](#), [48](#), [58](#), [58](#), [61](#), [62](#)
- `proc_freq`, [59](#)
- 
- `qflextable` (`flextable`), [35](#)
- 
- `read_pptx`, [56](#)
- `regulartable` (`flextable`), [35](#)
- `rotate`, [7](#), [18](#), [20](#), [28](#), [33](#), [38](#), [46](#), [55](#), [60](#), [72](#)
- 
- `save_as_html`, [15](#), [33](#), [40](#), [45](#), [48](#), [58](#), [59](#), [61](#), [62](#)
- `save_as_image`, [15](#), [33](#), [40](#), [45](#), [48](#), [58](#), [59](#), [61](#), [61](#)
- `set_caption`, [62](#)
- `set_footer_df` (`set_header_footer_df`), [64](#)
- `set_formatter`, [63](#)
- `set_formatter_type` (`set_formatter`), [63](#)
- `set_header_df` (`set_header_footer_df`), [64](#)
- `set_header_footer_df`, [5](#), [6](#), [64](#)
- `set_header_labels`, [65](#)
- `style`, [66](#)
- 
- `theme_alafoli`, [67](#), [67](#), [68–71](#)
- `theme_booktabs`, [67](#), [67](#), [68–71](#)
- `theme_box`, [67](#), [68](#), [69–71](#)
- 
- `theme_tron`, [67](#), [68](#), [68](#), [69–71](#)
- `theme_tron_legacy`, [67–69](#), [69](#), [70](#), [71](#)
- `theme_vader`, [67–69](#), [69](#), [70](#), [71](#)
- `theme_vanilla`, [67–70](#), [70](#), [71](#)
- `theme_zebra`, [67–70](#), [71](#)
- 
- `valign`, [7](#), [18](#), [20](#), [28](#), [33](#), [38](#), [46](#), [55](#), [60](#), [71](#)
- `vline`, [20–24](#), [42–44](#), [72](#), [73](#), [74](#)
- `vline_left`, [20–24](#), [42–44](#), [72](#), [73](#), [74](#)
- `vline_right`, [20–24](#), [42–44](#), [72](#), [73](#), [73](#)
- `void`, [74](#)
- 
- `width`, [17](#), [30](#), [31](#), [34](#), [37](#), [42](#), [56](#), [57](#), [75](#)
- 
- `xtable_to_flextable`, [75](#)