Package: rtables.officer (via r-universe)

November 26, 2024

Title Reporting Tables

Version 0.0.1.9008

Date 2024-11-26

Description This package is dedicated to convert `rtables` object to tables in use in `pptx` and `docx` files.

License Apache License 2.0 | file LICENSE

URL https://github.com/insightsengineering/rtables.officer, https://insightsengineering.github.io/rtables.officer/

BugReports https://github.com/insightsengineering/rtables.officer/issues

Depends formatters (>= 0.5.9), magrittr (>= 1.5), methods, R (>= 2.10), rtables (>= 0.6.10.9003)

Imports checkmate (>= 2.1.0), flextable (>= 0.9.6), lifecycle (>= 0.2.0), officer (>= 0.6.6), stats, stringi (>= 1.6)

Suggests broom (>= 1.0.6), car (>= 3.0-13), dplyr (>= 1.0.5), knitr (>= 1.42), r2rtf (>= 0.3.2), rmarkdown (>= 2.23), survival (>= 3.3-1), testthat (>= 3.0.4), tibble (>= 3.2.1), tidyr (>= 1.1.3), withr (>= 2.0.0), xml2 (>= 1.1.0)

VignetteBuilder knitr, rmarkdown

Remotes insightsengineering/rtables@main

Config/Needs/verdepcheck insightsengineering/formatters, insightsengineering/rtables, tidyverse/magrittr, mllg/checkmate, rstudio/htmltools, gagolews/stringi, tidymodels/broom, cran/car, tidyverse/dplyr, davidgohel/flextable, yihui/knitr, r-lib/lifecycle, davidgohel/officer, Merck/r2rtf, rstudio/rmarkdown, therneau/survival, r-lib/testthat, tidyverse/tibble, tidyverse/tidyr, r-lib/withr, r-lib/xml2

Config/testthat/edition 3

Encoding UTF-8

Language en-US

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Collate 'package.R' 'export_as_docx.R' 'as_flextable.R'

Config/pak/sysreqs libcairo2-dev libfontconfig1-dev libfreetype6-dev libfribidi-dev make libharfbuzz-dev libicu-dev libjpeg-dev libpng-dev libtiff-dev libxml2-dev libssl-dev

Repository https://pharmaverse.r-universe.dev

RemoteUrl https://github.com/insightsengineering/rtables.officer

RemoteRef HEAD

RemoteSha 9b0bb27b57223b9ec3c154317361ace1eb1b75e3

Contents

`export_as_docx` *Export as word document*

Description

From a table, produce a self-contained word document or attach it to a template word file (`template_file`). This function is based on the `tt_to_flextable()` transformer and the `officer` package.

Usage

```
export_as_docx(
  tt,
  file,
  doc_metadata = NULL,
  titles_as_header = FALSE,
  footers_as_text = TRUE,
  template_file = NULL,
  section_properties = section_properties_default(),
  ...
)

section_properties_default(
  page_size = c("letter", "A4"),
  orientation = c("portrait", "landscape")
)
```

`margins_potrait()`

`margins_landscape()`

Arguments

`tt`	(`TableTree` or related class) a `TableTree` object representing a populated table.
`file`	(`string`) string that indicates the final file output. Must have `.docx` extension.
`doc_metadata`	(`list` of `strings`) any value that can be used as metadata by `?officer::set_doc_properties`. Important text values are `title`, `subject`, `creator`, and `description`, while `created` is a date object.
`titles_as_header`	(`flag`) defaults to `TRUE` for `tt_to_flextable()`, so the table is self-contained as it makes additional header rows for `formatters::main_title()` string and `formatters::subtitles()` character vector (one per element). `FALSE` is suggested for `export_as_docx()`. This adds titles and subtitles as a text paragraph above the table. The same style is applied.
`footers_as_text`	(`flag`) defaults to `FALSE` for `tt_to_flextable()`, so the table is self-contained with the `flextable` definition of footnotes. `TRUE` is used for `export_as_docx()` to add the footers as a new paragraph after the table. The same style is applied, but with a smaller font.
`template_file`	(`string`) template file that `officer` will use as a starting point for the final document. Document attaches the table and uses the defaults defined in the template file.
`section_properties`	(`officer::prop_section`) an `officer::prop_section()` object which sets margins and page size. Defaults to `section_properties_default()`.
`...`	(`any`) additional arguments passed to `tt_to_flextable()`.
`page_size`	(`character(1)`) page size. Can be `"letter"` or `"A4"`. Defaults to `"letter"`.
`orientation`	(`character(1)`) page orientation. Can be `"portrait"` or `"landscape"`. Defaults to `"portrait"`.

Functions

- `section_properties_default()`: Helper function that defines standard portrait properties for tables.
- `margins_potrait()`: Helper function that defines standard portrait margins for tables.
- `margins_landscape()`: Helper function that defines standard landscape margins for tables.

Note

export_as_docx() has few customization options available. If you require specific formats and details, we suggest that you use tt_to_flextable() prior to export_as_docx. Only the titles_as_header and footer_as_text parameters must be re-specified if the table is changed first using tt_to_flextable().

See Also

tt_to_flextable()

Examples

```
lyt <- basic_table() %>%
  split_cols_by("ARM") %>%
  analyze(c("AGE", "BMRKR2", "COUNTRY"))

tbl <- build_table(lyt, ex_adsl)

# See how section_properties_portrait function is built for custom
tf <- tempfile(fileext = ".docx")
export_as_docx(tbl,
  file = tf,
  section_properties = section_properties_default(orientation = "landscape")
)
```

---

tt_to_flextable *Create a* flextable *from an* rtables *table*

Description

Principally used for export (export_as_docx()), this function produces a flextable from an rtables table. If theme = NULL, rtables-like style will be used. Otherwise, theme_docx_default() will produce a .docx-friendly table.

Usage

```
tt_to_flextable(
  tt,
  theme = theme_docx_default(),
  border = flextable::fp_border_default(width = 0.5),
  indent_size = NULL,
  titles_as_header = TRUE,
  bold_titles = TRUE,
  footers_as_text = FALSE,
  counts_in_newline = FALSE,
  paginate = FALSE,
  fontspec = NULL,
  lpp = NULL,
```

```
  cpp = NULL,
  ...,
  colwidths = NULL,
  tf_wrap = !is.null(cpp),
  max_width = cpp,
  total_page_height = 10,
  total_page_width = 10,
  autofit_to_page = TRUE
)

theme_docx_default(
  font = "Arial",
  font_size = 9,
  cell_margins = c(word_mm_to_pt(1.9), word_mm_to_pt(1.9), 0, 0),
  bold = c("header", "content_rows", "label_rows", "top_left"),
  bold_manual = NULL,
  border = flextable::fp_border_default(width = 0.5)
)

theme_html_default(
  font = "Courier",
  font_size = 9,
  cell_margins = 0.2,
  remove_internal_borders = "label_rows",
  border = flextable::fp_border_default(width = 1, color = "black")
)

word_mm_to_pt(mm)
```

Arguments

tt	(TableTree or related class) a TableTree object representing a populated table.
theme	(function or NULL) A theme function that is designed internally as a function of a flextable object to change its layout and style. If NULL, it will produce a table similar to rtables default. Defaults to theme_docx_default() that is a classic Word output. See details for more information.
border	(flextable::fp_border()) border style. Defaults to flextable::fp_border_default(width = 0.5).
indent_size	(numeric(1)) if NULL, the default indent size of the table (see formatters::matrix_form() indent_size, default is 2) is used. To work with docx, any size is multiplied by 1 mm (2.83 pt) by default.
titles_as_header	(flag) defaults to TRUE for tt_to_flextable(), so the table is self-contained as it makes additional header rows for formatters::main_title() string and formatters::subtitles()

character vector (one per element). FALSE is suggested for export_as_docx(). This adds titles and subtitles as a text paragraph above the table. The same style is applied.

bold_titles (flag or integer)
defaults to TRUE for tt_to_flextable(), so the titles are bold. If it is one or more integers, those lines will be bold.

footers_as_text
(flag)
defaults to FALSE for tt_to_flextable(), so the table is self-contained with the flextable definition of footnotes. TRUE is used for export_as_docx() to add the footers as a new paragraph after the table. The same style is applied, but with a smaller font.

counts_in_newline
(flag)
defaults to FALSE. In rtables text printing (formatters::toString()), the column counts, i.e. (N=xx), are always on a new line. For docx exports it could be necessary to print it on the same line.

paginate (flag)
when exporting .docx documents using export_as_docx, we suggest relying on the Microsoft Word pagination system. If TRUE, this option splits tt into different "pages" as multiple flextables. Cooperation between the two mechanisms is not guaranteed. Defaults to FALSE.

fontspec (font_spec)
a font_spec object specifying the font information to use for calculating string widths and heights, as returned by font_spec().

lpp (numeric(1))
maximum lines per page including (re)printed header and context rows.

cpp (numeric(1) or NULL)
width (in characters) of the pages for horizontal pagination. NA (the default) indicates cpp should be inferred from the page size; NULL indicates no horizontal pagination should be done regardless of page size.

... (any)
additional parameters to be passed to the pagination function. See rtables::paginate_table() for further details.

colwidths (numeric)
column widths for the resulting flextable(s). If NULL, the column widths estimated with formatters::propose_column_widths() will be used. When exporting into .docx these values are normalized to represent a fraction of the total_page_width. If these are specified, autofit_to_page is set to FALSE.

tf_wrap (flag)
whether the text for title, subtitles, and footnotes should be wrapped.

max_width (integer(1), string or NULL)
width that title and footer (including footnotes) materials should be word-wrapped to. If NULL, it is set to the current print width of the session (getOption("width")). If set to "auto", the width of the table (plus any table inset) is used. Parameter is ignored if tf_wrap = FALSE.

total_page_height
: (numeric(1))
total page height (in inches) for the resulting flextable(s). Used only to estimate number of lines per page (lpp) when paginate = TRUE. Defaults to 10.

total_page_width
: (numeric(1))
total page width (in inches) for the resulting flextable(s). Any values added for column widths is normalized by the total page width. Defaults to 10. If autofit_to_page = TRUE, this value is automatically set to the allowed page width.

autofit_to_page
: (flag)
defaults to TRUE. If TRUE, the column widths are automatically adjusted to fit the total page width. If FALSE, the colwidths are used as an indicative proportion of total_page_width. See flextable::set_table_properties(layout) for more details.

font
: (string)
defaults to "Arial". If the font is not available, flextable default is used. Please consider consulting the family column from systemfonts::system_fonts().

font_size
: (integer(1))
font size. Defaults to 9.

cell_margins
: (numeric(1) or numeric(4))
a numeric or a vector of four numbers indicating c("left", "right", "top", "bottom"). It defaults to 0 for top and bottom, and to 0.19 mm in word pt for left and right.

bold
: (character)
parts of the table text that should be in bold. Can be any combination of c("header", "content_rows", "label_rows", "top_left"). The first one renders all column names bold (not topleft content). The second and third option use formatters::make_row_df() to render content or/and label rows as bold.

bold_manual
: (named list or NULL)
list of index lists. See example for needed structure. Accepted groupings/names are c("header", "body").

remove_internal_borders
: (character)
defaults to "label_rows". Remove internal borders between rows. Currently there are no other options and can be turned off by providing any character value.

mm
: (numeric(1))
the value in mm to transform to pt.

Details

Themes can also be extended when you need only a minor change from a default style. You can either add your own theme to the theme call (e.g. c(theme_docx_default(), my_theme)) or create a new theme like shown in the examples. Please pay attention to the parameters' inputs as they are relevant for this to work properly. Indeed, it is possible to use some hidden values for building your own theme (hence the need of ...). In particular, tt_to_flextable sends in the

following variable: `tbl_row_class = rtables::make_row_df(tt)$node_class`. This is ignored if not used in the theme. See `theme_docx_default` for an example on own to retrieve these values and how to use them.

Value

A `flextable` object.

Functions

- `theme_docx_default()`: Main theme function for `export_as_docx()`.
- `theme_html_default()`: Theme function for html outputs.
- `word_mm_to_pt()`: Padding helper functions to transform mm to pt.

Note

Currently `cpp`, `tf_wrap`, and `max_width` are only used in pagination and do not yet have a clear cooperation with `colwidths` and `autofit_to_page`. at the moment it is suggested to use the `cpp` parameter family cautiously. If issues arise, please communicate with the maintainers or raise an issue.

See Also

`export_as_docx()`

`export_as_docx()`

Examples

```
analysisfun <- function(x, ...) {
  in_rows(
    row1 = 5,
    row2 = c(1, 2),
    .row_footnotes = list(row1 = "row 1 - row footnote"),
    .cell_footnotes = list(row2 = "row 2 - cell footnote")
  )
}

lyt <- basic_table(
  title = "Title says Whaaaat", subtitles = "Oh, ok.",
  main_footer = "ha HA! Footer!"
) %>%
  split_cols_by("ARM") %>%
  analyze("AGE", afun = analysisfun)

tbl <- build_table(lyt, ex_adsl)

# rtables style
tt_to_flextable(tbl, theme = NULL)

tt_to_flextable(tbl, theme = theme_docx_default(font_size = 6))
```

```
# Example with multiple themes (only extending the docx default!)
my_theme <- function(x, ...) {
 flextable::border_inner(x, part = "body", border = flextable::fp_border_default(width = 0.5))
}
flx <- tt_to_flextable(tbl, theme = c(theme_docx_default(), my_theme))

# Custom theme
special_bold <- list(
  "header" = list("i" = 1, "j" = c(1, 3)),
  "body" = list("i" = c(1, 2), "j" = 1)
)
custom_theme <- theme_docx_default(
  font_size = 10,
  font = "Brush Script MT",
  border = flextable::fp_border_default(color = "pink", width = 2),
  bold = NULL,
  bold_manual = special_bold
)
tt_to_flextable(tbl,
  border = flextable::fp_border_default(color = "pink", width = 2),
  theme = custom_theme
)

# Extending themes
my_theme <- function(font_size = 6) { # here can pass additional arguments for default theme
  function(flx, ...) {
    # First apply theme_docx_default
    flx <- theme_docx_default(font_size = font_size)(flx, ...)

    # Then apply additional styling
    flx <- flextable::border_inner(flx,
      part = "body",
      border = flextable::fp_border_default(width = 0.5)
    )

    return(flx)
  }
}
flx <- tt_to_flextable(tbl, theme = my_theme())
```

Index