Package: officerExtras (via r-universe)

September 18, 2024

Title Extra Helpers for 'officer'

Version 0.0.1.9002

Description Helper and convenience functions using the 'officer' package to modify docx files.

License MIT + file LICENSE

URL https://github.com/elipousson/officerExtras, https://elipousson.github.io/officerExtras/

BugReports https://github.com/elipousson/officerExtras/issues

Imports cli, glue, lifecycle, officer, rlang (>= 1.1.0), tibble, utils, vctrs, xml2

Suggests covr, ggplot2, gt (>= 0.7.0), rmarkdown, testthat (>= 3.0.0), withr

Config/testthat/edition 3

Encoding UTF-8

Language en

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

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

RemoteUrl https://github.com/elipousson/officerExtras

RemoteRef HEAD

RemoteSha 32fa9a1ca097917b1988afa845921badb2353502

Contents

`add_list_to_body` *Add a formatted list of items to a* `rdocx` *object*

Description

This function assumes that the input docx object has a list style. Use `read_docx_ext()` with `allow_null = TRUE` to create a new rdocx object with the "List Bullet" and "List Number" styles. "List Paragraph" can be used but the resulting formatted list will not have bullets or numbers.

Usage

```
add_list_to_body(
  docx,
  values,
  style = "List Bullet",
  keep_na = FALSE,
  before = NULL,
  after = "",
  ...
)
```

Arguments

docx	A rdocx object.
values	Character vector with values for list.
style	Style to use for list of values, Default: 'List Bullet'
keep_na	If TRUE, keep NA values in values Default: FALSE
before	Value to insert before the list, Default: NULL
after	Value to insert after the list, Default: ''
...	Additional parameters passed to cursor_docx()

Value

A modified rdocx object.

See Also

add_to_body(), vec_add_to_body()

---

add_to_body *Add a xml string, text paragraph, or gt object at a specified position in a rdocx object*

Description

Wrappers for officer::body_add_par(), officer::body_add_gg(), and officer::body_add_xml() that use the cursor_docx() helper function to allow users to pass the value and keyword, id, or index value used to place a "cursor" within the document using a single function. If pos = NULL, add_to_body() calls officer::body_add() instead of officer::body_add_par(). If value is a gt_tbl object, value is passed as the gt_object parameter for add_gt_to_body().

- add_text_to_body() passes value to glue::glue() to add support for glue string interpolation.
- add_gt_to_body() converts gt tables to OOXML with gt::as_word().
- add_gg_to_body() adds a caption following the plots using the labels from the plot object.

Usage

```
add_to_body(
  docx,
  keyword = NULL,
  id = NULL,
  index = NULL,
  value = NULL,
  str = NULL,
  style = NULL,
  pos = "after",
```

```
  ...,
  gt_object = NULL,
  call = caller_env()
)

add_text_to_body(
  docx,
  value,
  style = NULL,
  pos = "after",
  .na = "NA",
  .null = NULL,
  .envir = parent.frame(),
  ...
)

add_xml_to_body(docx, str, pos = "after", ..., call = caller_env())

add_gt_to_body(
  docx,
  gt_object,
  align = "center",
  caption_location = c("top", "bottom", "embed"),
  caption_align = "left",
  split = FALSE,
  keep_with_next = TRUE,
  pos = "after",
  tablecontainer = TRUE,
  ...,
  call = caller_env()
)

add_gg_to_body(
  docx,
  value,
  caption = "title",
  caption_style = style,
  autonum = NULL,
  style = "Normal",
  pos = "after",
  ...
)

add_value_with_keys(docx, value, ..., .f = add_text_to_body)

add_str_with_keys(docx, str, ..., .f = add_xml_to_body)
```

Arguments

docx	A rdocx object.
keyword, id	A keyword string used to place cursor with officer::cursor_reach() or bookmark id with officer::cursor_bookmark(). Defaults to NULL. If keyword or id are not provided, the gt object is inserted at the front of the document.
index	A integer matching a doc_index value appearing in a summary of the docx object created with officer::docx_summary(). If index is for a paragraph value, the text of the pargraph is used as a keyword.
value	object to add in the document. Supported objects are vectors, data.frame, graphics, block of formatted paragraphs, unordered list of formatted paragraphs, pretty tables with package flextable, 'Microsoft' charts with package mschart.
str	a wml string
style	paragraph style name. These names are available with function styles_info and are the names of the Word styles defined in the base document (see argument path from read_docx).
pos	where to add the new element relative to the cursor, one of "after", "before", "on".
...	Additional parameters passed to officer::body_add_par(), officer::body_add_gg(), or officer::body_add().
gt_object	A gt object converted to an OOXML string with gt::as_word() then passed to add_xml_to_body() as str parameter. Required for add_gt_to_body().
call	The execution environment of a currently running function, e.g. caller_env(). The function will be mentioned in error messages as the source of the error. See the call argument of abort() for more information.
.na	[character(1): 'NA'] Value to replace NA values with. If NULL missing values are propagated, that is an NA result will cause NA output. Otherwise the value is replaced by the value of .na.
.null	[character(1): 'character()'] Value to replace NULL values with. If character() whole output is character(). If NULL all NULL values are dropped (as in paste0()). Otherwise the value is replaced by the value of .null.
.envir	[environment: parent.frame()] Environment to evaluate each expression in. Expressions are evaluated from left to right. If .x is an environment, the expressions are evaluated in that environment and .envir is ignored. If NULL is passed, it is equivalent to emptyenv().
align	*Table alignment* scalar<character> // *default:* "center" An option for table alignment. Can either be "center", "left", or "right".
caption_location	*Caption location* singl-kw:[top|bottom|embed] // *default:* "top" Determines where the caption should be positioned. This can either be "top", "bottom", or "embed".

caption_align *Caption alignment*

Determines the alignment of the caption. This is either "left" (the default), "center", or "right". This option is only used when caption_location is not set as "embed".

split *Allow splitting of a table row across pages*
scalar<logical> // *default:* FALSE

A logical value that indicates whether to activate the Word option Allow row to break across pages.

keep_with_next *Keeping rows together*
scalar<logical> // *default:* TRUE

A logical value that indicates whether a table should use Word option Keep rows together.

tablecontainer If TRUE (default), add tables inside of a tablecontainer tag that automatically adds a table number and converts the gt title into a table caption. This feature is based on code from the gto package by Ellis Hughes to transform the gt_object to OOXML and insert the XML into the docx object.

caption Name of the ggplot2 label to use as a caption if plot passed to value has a label for this value. Defaults to "title".

caption_style Passed to style for officer::body_add_caption(). Defaults to same value as style.

autonum an object generated with function run_autonum

.f Any function that takes a docx and value parameter and returns a rdocx object. A keyword parameter must also be supported if named is TRUE. Defaults to add_text_to_body().

Details

Using add_value_with_keys() or add_str_with_keys()

add_value_with_keys() supports value vectors of length 1 or longer. If value is named, the names are assumed to be keywords indicating the cursor position for adding each value in the vector. If value is not named, a keyword parameter with the same length as value must be provided. When named = FALSE, no keyword parameter is required. Add add_str_with_keys() works identically but uses a str parameter and .f defaults to add_xml_to_body().

Note, as of July 2023, both add_value_with_keys() and add_str_with_keys() are superseded by vec_add_to_body().

Value

A rdocx object with xml, gt tables, or paragraphs of text added.

Author(s)

Ellis Hughes <ellis.h.hughes@gsk.com> (ORCID)

Examples

```
if (rlang::is_installed("gt")) {
  docx_example <- read_docx_ext(
    filename = "example.docx",
    path = system.file("doc_examples", package = "officer")
  )

  tab_1 <-
    gt::gt(
      gt::exibble,
      rowname_col = "row",
      groupname_col = "group"
    )

  add_gt_to_body(
    docx_example,
    tab_1,
    keyword = "Title 1"
  )

  tab_str <- gt::as_word(tab_1)

  add_str_with_keys(
    docx_example,
    str = c("Title 1" = tab_str, "Title 2" = tab_str)
  )
}
```

---

check_officer *Check if x is a rdocx, rpptx, or rxlsx object*

Description

Check if x is a rdocx, rpptx, or rxlsx object

Usage

```
check_officer(
  x,
  arg = caller_arg(x),
  what = c("rdocx", "rpptx", "rxlsx"),
  call = caller_env(),
  ...
)

check_docx(x, arg = caller_arg(x), call = caller_env(), ...)

check_pptx(x, arg = caller_arg(x), call = caller_env(), ...)
```

```
check_xlsx(x, arg = caller_arg(x), call = caller_env(), ...)

check_block_list(
  x,
  arg = caller_arg(x),
  allow_empty = FALSE,
  allow_null = FALSE,
  call = caller_env()
)
```

Arguments

x	Object to check.
arg	Argument name of object to check. Used to improve cli::cli_abort() messages. Defaults to caller_arg(x).
what	Class names to check
call	The execution environment of a currently running function, e.g. caller_env(). The function will be mentioned in error messages as the source of the error. See the call argument of abort() for more information.
...	Additional parameters passed to cli::cli_abort()
allow_empty	If TRUE, check_block_list() allows an empty block list. Defaults to FALSE.
allow_null	If FALSE (default), error if x is NULL.

---

check_officer_summary *Check a summary data.frame created from a rdocx or rpptx object*

Description

Check an object and error if an object is not a data.frame with the required column names to be a summary data.frame created from a rdocx or rpptx object. Optionally check for the number of rows, a specific content_type value, or if tables are included in the document the summary was created from.

Usage

```
check_officer_summary(
  x,
  n = NULL,
  content_type = NULL,
  summary_type = "doc",
  tables = FALSE,
  ...,
  arg = caller_arg(x),
  call = parent.frame()
)
```

Arguments

x	An object to check if it is a data.frame object created with officer::docx_summary() or another summary function.
n	Required number of rows. Optional. If n is more than length 1, checks to make sure the number of rows is within the range of max(n) and min(n). Defaults to NULL.
content_type	Required content_type, e.g. "paragraph", "table cell", or "image". Optional. Defaults to NULL.
summary_type	Summary type. Options "doc", "docx", "pptx", "slide", or "layout". "doc" requires the data.frame include a "content_type" column but allows columns for either a docx or pptx summary.
tables	If TRUE, require that the summary include the column names indicated a table is present in the rdocx or rpptx summary.
...	Additional parameters passed to cli::cli_abort()
arg	Argument name to use in error messages. Defaults to caller_arg(x)
call	The execution environment of a currently running function, e.g. call = caller_env(). The corresponding function call is retrieved and mentioned in error messages as the source of the error. You only need to supply call when throwing a condition from a helper function which wouldn't be relevant to mention in the message. Can also be NULL or a defused function call to respectively not display any call or hard-code a code to display. For more information about error calls, see Including function calls in error messages.

See Also

Other summary functions: officer_summary(), officer_summary_levels()

---

check_office_fileext *Check file path for a docx, pptx, or xlsx file extension*

Description

Check file path for a docx, pptx, or xlsx file extension

Usage

```
check_office_fileext(
  x,
  arg = caller_arg(x),
  fileext = c("docx", "pptx", "xlsx"),
  allow_null = FALSE,
  call = caller_env(),
```

```
  ...
)

check_docx_fileext(x, arg = caller_arg(x), call = caller_env(), ...)

check_pptx_fileext(x, arg = caller_arg(x), call = caller_env(), ...)

check_xlsx_fileext(x, arg = caller_arg(x), call = caller_env(), ...)
```

Arguments

x	Object to check.
arg	Argument name of object to check. Used to improve cli::cli_abort() messages. Defaults to caller_arg(x).
fileext	File extensions to allow without error. Defaults to "docx", "pptx", "xlsx".
allow_null	If TRUE and x is NULL, invisibly return NULL. If FALSE (default), error unless x is a character vector with file extension matching the supplied fileext value.
call	The execution environment of a currently running function, e.g. caller_env(). The function will be mentioned in error messages as the source of the error. See the call argument of abort() for more information.
...	Additional parameters passed to cli::cli_abort()

---

combine_docx *Combine multiple rdocx objects or docx files*

Description

combine_docx() is a variant of officer::body_add_docx() that allows any number of input files and supports rdocx objects as well as Word file paths. Optionally use a separator between files or objects.

Please note that when you create a new rdocx object with this function (or officer::body_add_docx()) the added content will not appear in a summary data frame created with officer_summary() and is not accessible to other functions until the document is *opened and edited* with Microsoft Word. This is part of how the OOXML AltChunk Class works and can't be avoided.

Usage

```
combine_docx(
  ...,
  docx = NULL,
  .list = list2(...),
  pos = "after",
  sep = NULL,
  call = caller_env()
)
```

Arguments

...	Any number of additional rdocx objects or docx file paths.
docx	A rdocx object or a file path with a docx file extension. Defaults to NULL.
.list	Any number of additional rdocx objects or docx file paths passed as a list. Defaults to list2()
pos	where to add the new element relative to the cursor, one of "after", "before", "on".
sep	Separator to use docx files. A bare function that takes a rdocx object as the only parameter, such as officer::body_add_break or another object passed to add_to_body() as the value parameter. Optional. Defaults to NULL.
call	The execution environment of a currently running function, e.g. caller_env(). The function will be mentioned in error messages as the source of the error. See the call argument of abort() for more information.

Value

A rdocx object.

See Also

officer::body_add_docx()

Examples

```
## Not run:
if (interactive()) {
  docx_path <- system.file("doc_examples", "example.docx", package = "officer")
  docx <- read_officer(docx_path)

  combine_docx(
    docx,
    docx_path
  )
}

## End(Not run)
```

---

convert_docx *Convert an rodcx object or Word file to another format with pandoc*

Description

convert_docx() uses rmarkdown::pandoc_convert() to convert a rodcx object or Word file to another format with pandoc. If the "to" parameter contains a file extension, it is assumed to be an output file name. If you want to convert a file to a Word document, use the input parameter for the path to the Markdown, HTML, or other file.

Usage

```
convert_docx(
  docx = NULL,
  to = NULL,
  input = NULL,
  output = NULL,
  path = NULL,
  options = NULL,
  extract_media = TRUE,
  overwrite = TRUE,
  quiet = TRUE,
  ...
)
```

Arguments

docx	A rdocx object or path to a docx file. Optional if input is provided.
to	Format to convert to (if not specified, you must specify output)
input	Character vector containing paths to input files (files must be UTF-8 encoded)
output	Output file (if not specified then determined based on format being converted to).
path	File path for converted file passed to wd parameter of rmarkdown::pandoc_convert(). If docx is a rdocx object, path defaults to current working directory instead of the base directory of the input file path.
options	Character vector of command line options to pass to pandoc.
extract_media	If TRUE, append "--extract-media=." to options. Defaults to FALSE.
overwrite	If TRUE (default), remove file at path if it already exists. If FALSE and file exists, this function aborts.
quiet	If TRUE, suppress alert messages and pass FALSE to verbose parameter of rmarkdown::pandoc_convert() Defaults TRUE.
...	Arguments passed on to rmarkdown::pandoc_convert from Format to convert from (if not specified then the format is determined based on the file extension of input). citeproc TRUE to run the pandoc-citeproc filter (for processing citations) as part of the conversion.

Details

Supported input and output formats are described in the pandoc user guide.

The system path as well as the version of pandoc shipped with RStudio (if running under RStudio) are scanned for pandoc and the highest version available is used.

Value

Executes a call to pandoc using rmarkdown::pandoc_convert() to create a file from an officer object or a docx file.

See Also

rmarkdown::pandoc_convert()

Examples

```
docx_example <- read_officer(
  system.file("doc_examples/example.docx", package = "officer")
)

convert_docx(
  docx_example,
  to = "markdown"
)

withr::with_tempdir({
  convert_docx(
    docx_example,
    output = "docx_example.html"
  )
})
```

---

cursor_docx *Set cursor position in rdocx object based on keyword, id, or index*

Description

A combined function for setting cursor position with officer::cursor_reach(), officer::cursor_bookmark(), or using a doc_index value from officer::docx_summary(). Defaults to using officer::cursor_end(), officer::cursor_begin(), officer::cursor_backward(), or officer::cursor_forward() if keyword, id, and index are all NULL.

Usage

```
cursor_docx(
  docx,
  keyword = NULL,
  id = NULL,
  index = NULL,
  default = "end",
  quiet = FALSE,
  call = caller_env()
)
```

Arguments

docx	A rdocx object.

keyword, id	A keyword string used to place cursor with officer::cursor_reach() or bookmark id with officer::cursor_bookmark(). Defaults to NULL. If keyword or id are not provided, the gt object is inserted at the front of the document.
index	A integer matching a doc_index value appearing in a summary of the docx object created with officer::docx_summary(). If index is for a paragraph value, the text of the pargraph is used as a keyword.
default	Character string with one of the following options: c("end", "begin", "backward", "forward") to set cursor position. Only used if keyword, id, and index are all NULL.
quiet	If FALSE (default) warn when keyword is not found.
call	The execution environment of a currently running function, e.g. caller_env(). The function will be mentioned in error messages as the source of the error. See the call argument of abort() for more information.

See Also

officer::cursor_begin(), officer::docx_summary()

---

dims_docx_ext *Get page and block layout dimensions from an rdocx object*

Description

This function extends officer::docx_dim() by also returning the body text dimensions within the margins, the aspect ratio of the page and body, and the page orientation as a string ("landscape" or "portrait").

Usage

```
dims_docx_ext(docx)
```

Arguments

docx	A rdocx object to get dimensions for.

See Also

officer::docx_dim()

`fill_with_pattern` *Fill a new column based on an existing column depending on a pattern*

Description

This function uses `vctrs::vec_fill_missing()` to convert hierarchically nested headings and text into a rectangular data.frame. It is an experimental function that may be modified or removed. At present, it is only used by `officer_tables()`.

Usage

```
fill_with_pattern(
  x,
  pattern = "^heading",
  pattern_col = "style_name",
  fill_col = "text",
  col = "heading",
  direction = c("down", "up", "downup", "updown"),
  call = caller_env()
)
```

Arguments

`x`	A input data.frame (assumed to be from `officer_summary()` for default values).
`pattern`	Passed to `grepl()` as the pattern identifying which rows of the fill_col should have values pulled into the new column named by col. Defaults to "^heading" which matches the default heading style names.
`pattern_col`	Name of column to use for pattern matching, Defaults to "style_name".
`fill_col`	Name of column to fill , Defaults to "text".
`col`	Name of new column to fill with values from fill_col, Default: Defaults to "heading"
`direction`	Direction of fill passed to `vctrs::vec_fill_missing()`, Default: c("down", "up", "downup", "updown")
`call`	The execution environment of a currently running function, e.g. `caller_env()`. The function will be mentioned in error messages as the source of the error. See the `call` argument of `abort()` for more information.

Value

A data.frame with an additional column taking the name from col and the values from the column named in fill_col.

See Also

`vctrs::vec_fill_missing()`

is_officer *Is this object an officer class object?*

Description

`is_officer()` and variants are basic wrappers for `inherits()` to check for object classes created by functions from the officer package. These functions are intended for internal use but are exposed for use by other extension developers.

Usage

```
is_officer(x, what = c("rdocx", "rpptx", "rxlsx"))

is_rdocx(x)

is_rpptx(x)

is_block_list(x)
```

Arguments

x	A object to test
what	Class or classes passed to `inherits()`

make_block_list *Make a list of blocks to add to a Word document or PowerPoint presentation*

Description

[Experimental]

`make_block_list()` extends `officer::block_list()` by supporting a list of inputs and optionally combining parameters with an existing block list (using the blocks parameter). Unlike `officer::block_list()`, `make_block_list()` errors if no input parameters are provided.

`combine_blocks()` takes any number of `block_list` objects and combined them into a single `block_list`. Both functions are not yet working as expected.

Usage

```
make_block_list(blocks = NULL, ..., allow_empty = FALSE, call = caller_env())

combine_blocks(...)
```

Arguments

blocks	A list of parameters to pass to officer::block_list() or a block_list object. If parameters are provided to both ... and blocks is a block_list, the additional parameters are appended to the end of blocks.
...	For make_block_list(), these parameters are passed to officer::block_list() and must *not* include block_list objects. For combine_blocks(), these parameters must all be block_list objects.
allow_empty	If TRUE, check_block_list() allows an empty block list. Defaults to FALSE.
call	The execution environment of a currently running function, e.g. caller_env(). The function will be mentioned in error messages as the source of the error. See the call argument of abort() for more information.

See Also

Other block list functions: officer_add_blocks()

---

officer_add_blocks *Add a list of blocks into a Word document or PowerPoint presentation*

Description

[Experimental]

officer_add_blocks() supports adding a list of blocks to rdocx (using add_blocks_to_body()) or rpptx objects (using officer::ph_with()). add_blocks_to_body() is a variant of officer::body_add_blocks() that allows users to set the cursor position before adding the block list using cursor_docx().

Usage

```
officer_add_blocks(
  x,
  blocks,
  pos = "after",
  location = NULL,
  level_list = integer(0),
  ...,
  call = caller_env()
)

add_blocks_to_body(
  docx,
  blocks,
  pos = "after",
  keyword = NULL,
  id = NULL,
  index = NULL,
  ...
)
```

Arguments

x	A rdocx or rpptx object. Required.
blocks	set of blocks to be used as footnote content returned by function block_list().
pos	where to add the new element relative to the cursor, one of "after", "before", "on".
location	If NULL, location defaults to officer::ph_location_type()
level_list	The list of levels for hierarchy structure as integer values. If used the object is formated as an unordered list. If 1 and 2, item 1 level will be 1, item 2 level will be 2.
...	further arguments passed to or from other methods. When adding a ggplot object or plot_instr, these arguments will be used by png function.
call	The execution environment of a currently running function, e.g. caller_env(). The function will be mentioned in error messages as the source of the error. See the call argument of abort() for more information.
docx	A rdocx object.
keyword, id	A keyword string used to place cursor with officer::cursor_reach() or bookmark id with officer::cursor_bookmark(). Defaults to NULL. If keyword or id are not provided, the gt object is inserted at the front of the document.
index	A integer matching a doc_index value appearing in a summary of the docx object created with officer::docx_summary(). If index is for a paragraph value, the text of the pargraph is used as a keyword.

See Also

Other block list functions: make_block_list()

---

officer_media *Copy media from a docx or pptx file or a rdocx or rpptx object to a target folder*

Description

Unzip a docx or pptx file to a temporary directory, check if the directory contains a media folder, and copy media files to the directory set by dir. If a rdocx or rpptx object is provided, files are copied from the temporary package_dir associated with the object (accessible via x[["package_dir"]]).

Usage

```
officer_media(
  filename = NULL,
  path = NULL,
  x = NULL,
  target = "media",
  list = FALSE,
  overwrite = TRUE
)
```

Arguments

filename, path	File name and path for a docx or pptx file. One of the two must be provided. Ignored if x is provided. Defaults to NULL.
x	A rdocx or rpptx object containing one or more media file.
target	Folder name or path to copy media files. dir is created if no folder exists at that location.
list	If TRUE, display a message listing files contained in the docx or pptx file but do not copy the files to dir. Defaults to FALSE.
overwrite	If TRUE (default), overwrite any files with the same names at target path. If FALSE, abort if files with the same names already exist.

See Also

officer::media_extract()

Examples

```
officer_media(
  system.file("doc_examples/example.pptx", package = "officer"),
  list = TRUE
)
```

---

officer_properties *Get doc properties for a rdocx or rpptx object as a list*

Description

officer_properties() is a variant on officer::doc_properties() that will warn instead of error if document properties can't be found

Usage

```
officer_properties(x, values = list(), keep.null = FALSE, call = caller_env())
```

Arguments

x	A rdocx or rpptx object.
values	A named list with new properties to replace existing document properties before they are returned as a named list.
keep.null	Passed to utils::modifyList(). If TRUE, retain properties in returned list even if they have NULL values.
call	The execution environment of a currently running function, e.g. caller_env(). The function will be mentioned in error messages as the source of the error. See the call argument of abort() for more information.

Value

A named list of existing document properties or (if values is supplied) modified document properties.

`officer_summary` *Summarize a rdocx or rpptx object*

Description

`officer_summary()` extends `officer::docx_summary()` and other officer summary functions by handling multiple input types within a single function. The preserve parameter is supported by officer version >= 0.6.3 (currently the development version) and it is ignored unless a minimum supported version of officer is installed.

Usage

```
officer_summary(
  x,
  summary_type = "doc",
  index = NULL,
  preserve = FALSE,
  as_tibble = TRUE,
  call = caller_env()
)
```

Arguments

`x`	A rdocx or rpptx object passed to `officer::docx_summary()`, `officer::pptx_summary()`, `officer::slide_summary()`, or `officer::layout_summary()`. If x is a data frame created with one of those functions, it is returned as is (this feature may be removed in the future).
`summary_type`	Summary type. Options "doc", "docx", "pptx", "slide", or "layout". "doc" requires the data.frame include a "content_type" column but allows columns for either a docx or pptx summary.
`index`	slide index
`preserve`	If `FALSE` (default), text in table cells is collapsed into a single line. If `TRUE`, line breaks in table cells are preserved as a "\n" character. This feature is adapted from `docxtractr::docx_extract_tbl()` published under a MIT licensed in the `{docxtractr}` package by Bob Rudis.
`as_tibble`	If `TRUE` (default), return a tibble data frame.
`call`	The execution environment of a currently running function, e.g. `call = caller_env()`. The corresponding function call is retrieved and mentioned in error messages as the source of the error. You only need to supply `call` when throwing a condition from a helper function which wouldn't be relevant to mention in the message.

Can also be NULL or a defused function call to respectively not display any call or hard-code a code to display.

For more information about error calls, see Including function calls in error messages.

Value

A tibble or data frame object.

See Also

Other summary functions: check_officer_summary(), officer_summary_levels()

Examples

```
docx_example <- read_officer(
  system.file("doc_examples", "example.docx", package = "officer")
)

officer_summary(docx_example)

pptx_example <- read_officer(
  "example.pptx", system.file("doc_examples", package = "officer")
)

officer_summary(pptx_example)

officer_summary(pptx_example, "slide", 1)
```

---

officer_summary_levels

Create new columns to an officer summary data.frame based on specific levels

---

Description

This function works with fill_with_pattern() to help convert hierarchically organized text, e.g. text with leveled headings or other styles into a data.frame where new columns hold the value of the preceding (or succeeding) heading text.

Usage

```
officer_summary_levels(
  x,
  levels = NULL,
  levels_from = "style_name",
  exclude_levels = NULL,
```

```
  fill_col = "text",
  direction = c("down", "up", "downup", "updown"),
  ...,
  call = caller_env()
)
```

Arguments

x	A input data.frame (assumed to be from officer_summary() for default values).
levels	Levels to use. If NULL (default), levels is set to unique, non-NA values from the levels_from column.
levels_from	Column name to use for identifying levels. Defaults to "style_name".
exclude_levels	Levels to exclude from process of adding new columns.
fill_col	Name of column to fill , Defaults to "text".
direction	Direction of fill passed to vctrs::vec_fill_missing(), Default: c("down", "up", "downup", "updown")
...	Arguments passed on to officer_summary as_tibble If TRUE (default), return a tibble data frame. summary_type Summary type. Options "doc", "docx", "pptx", "slide", or "layout". "doc" requires the data.frame include a "content_type" column but allows columns for either a docx or pptx summary. index slide index preserve If FALSE (default), text in table cells is collapsed into a single line. If TRUE, line breaks in table cells are preserved as a "\n" character. This feature is adapted from docxtractr::docx_extract_tbl() published under a MIT licensed in the {docxtractr} package by Bob Rudis.
call	The execution environment of a currently running function, e.g. caller_env(). The function will be mentioned in error messages as the source of the error. See the call argument of abort() for more information.

Value

A summary data frame with additional columns based on the supplied levels.

See Also

Other summary functions: check_officer_summary(), officer_summary()

`officer_tables` *Get tables from a rdocx or rpptx object*

Description

Get one or more tables from a rdocx or rpptx object. `officer_tables()` returns a list of data frames and `officer_table()` returns a single table as a data frame. These functions are based on example code on extracting Word document and PowerPoint slides in the officeverse documentation. Some additional features including the type_convert parameter and the addition of doc_index values as the default names for the returned list of tables are based on this blog post by Matt Dray.

Usage

```
officer_tables(
  x,
  index = NULL,
  has_header = TRUE,
  col = NULL,
  preserve = FALSE,
  ...,
  stack = FALSE,
  type_convert = FALSE,
  nm = NULL,
  call = caller_env()
)

officer_table(
  x,
  index = NULL,
  has_header = TRUE,
  col = NULL,
  ...,
  call = caller_env()
)
```

Arguments

`x`	A rdocx or rpptx object or a data frame created with `officer_summary()`.
`index`	A index value matching a doc_index value for a table in the summary data frame, Default: `NULL`
`has_header`	If `TRUE` (default), tables are expected to have implicit headers even if the Word table does not have an explicit header row. If `FALSE`, only explicit header rows will be used as column names.
`col`	If col is supplied, `officer_table()` passes col and the additional parameters in ... to `fill_with_pattern()`. This allows the addition of preceding headings or captions as a column within the data.frame returned by `officer_tables()`.

This is an experimental feature and may be modified or removed. Defaults to NULL.

preserve — If FALSE (default), text in table cells is collapsed into a single line. If TRUE, line breaks in table cells are preserved as a "\n" character. This feature is adapted from docxtractr::docx_extract_tbl() published under a MIT licensed in the {docxtractr} package by Bob Rudis.

... — Additional parameters passed to fill_with_pattern().

stack — If TRUE and all tables share the same number of columns, return a single combined data frame instead of a list. Defaults to FALSE.

type_convert — If TRUE, convert columns for the returned data frames to the appropriate type using utils::type.convert().

nm — Names to use for returned list of tables. If NULL (default), the names are set to the doc_index values using the pattern "doc_index_<doc_index_number>".

call — The execution environment of a currently running function, e.g. call = caller_env(). The corresponding function call is retrieved and mentioned in error messages as the source of the error.

You only need to supply call when throwing a condition from a helper function which wouldn't be relevant to mention in the message.

Can also be NULL or a defused function call to respectively not display any call or hard-code a code to display.

For more information about error calls, see Including function calls in error messages.

Value

A list of data frames or, if stack is TRUE, a single data frame.

See Also

docxtractr::docx_extract_all()

Examples

```
docx_example <- read_docx_ext(
  filename = "example.docx",
  path = system.file("doc_examples", package = "officer")
)

officer_tables(docx_example)

pptx_example <- read_pptx_ext(
  filename = "example.pptx",
  path = system.file("doc_examples", package = "officer")
)

officer_tables(pptx_example)[[1]]
```

read_docs_url — *Read a Google Docs document, Slides, or Sheets URL to a rdocx, rpptx, or rxlsx object*

Description

Uses the Google Docs, Slides, or Sheets URL to export a file locally, read to an officer object. If filename is NULL, exported file is removed after export.

Usage

```
read_docs_url(url, format = NULL, filename = NULL, path = NULL, quiet = TRUE)
```

Arguments

url	A URL for a Google Doc, Google Slides presentation, or Google Sheets.
format	File format to use for export URL (typically set automatically). Options are NULL (default), "doc", "pptx", or "xslx". "pdf" and "csv" be supported in the future.
filename	Destination file name. Optional. If filename is NULL, downloaded file is removed as part of the function execution.
path	Folder path. Optional.
quiet	If TRUE, suppress messages when downloading file.

See Also

read_officer()

read_officer — *Read a docx, pptx, potx, or xlsx file or use an existing object from officer if provided*

Description

read_officer() is a variant of officer::read_docx(), officer::read_pptx(), and officer::read_xlsx() that allows users to read different Microsoft Office file types with a single function. read_docx_ext(), read_pptx_ext(), and read_xlsx_ext() are wrappers for read_officer() that require the matching input file type. All versions allow both a filename and path (the officer functions only use a path). If a rdocx, rpptx, or rxlsx class object is provided to x, the object is checked based on the fileext parameter and then returned as is.

Usage

```
read_officer(
  filename = NULL,
  path = NULL,
  fileext = c("docx", "pptx", "xlsx"),
  x = NULL,
  arg = caller_arg(x),
  allow_null = TRUE,
  quiet = TRUE,
  call = parent.frame(),
  ...
)

read_docx_ext(
  filename = NULL,
  path = NULL,
  docx = NULL,
  allow_null = FALSE,
  quiet = TRUE
)

read_pptx_ext(
  filename = NULL,
  path = NULL,
  pptx = NULL,
  allow_null = FALSE,
  quiet = TRUE
)

read_xlsx_ext(
  filename = NULL,
  path = NULL,
  xlsx = NULL,
  allow_null = FALSE,
  quiet = TRUE
)
```

Arguments

filename, path	File name and path. Default: NULL. Must include a "docx", "pptx", or "xlsx" file path. "dotx" and "potx" files are also supported.
fileext	File extensions to allow without error. Defaults to "docx", "pptx", "xlsx".
x	A rdocx, rpptx, or rxlsx class object If x is provided, filename and path are ignored. Default: NULL
arg	Argument name of object to check. Used to improve cli::cli_abort() messages. Defaults to caller_arg(x).

allow_null	If TRUE, function supports the default behavior of officer::read_docx(), officer::read_pptx(), or officer::read_xlsx() and returns an empty document if x, filename, and path are all NULL. If FALSE, one of the three parameters must be supplied.
quiet	If FALSE, warn if docx is provided when filename and/or path are also provided. Default: TRUE.
call	The execution environment of a currently running function, e.g. caller_env(). The function will be mentioned in error messages as the source of the error. See the call argument of abort() for more information.
...	Additional parameters passed to cli::cli_abort()
docx, pptx, xlsx	A rdocx, rpptx, or rxlsx class object passed to the x parameter of read_officer() by the variant functions. Defaults to NULL.

Value

A rdocx, rpptx, or rxlsx object.

See Also

officer::read_docx()

---

reduce_officer *Reduce a list to a single officer object*

Description

reduce_officer() is a wrapper for purrr::reduce().

Usage

```
reduce_officer(
  x = NULL,
  .f = function(x, value, ...) {
     vec_add_to_body(x, value = value, ...)
 },
  value = NULL,
  ...,
  .path = NULL
)
```

Arguments

x	File path or officer object.
.f	Any function taking an officer object as the first parameter and a value as the second parameter, Defaults to anonymous function: function(x, value, ...) { vec_add_to_body(x, value = value, ...) }
value	A vector of values that are support by the function passed to .f, Default: NULL

...	Additional parameters passed to purrr::reduce().
.path	If .path not NULL, it should be a file path that is passed to write_officer(), allowing you to modify a docx file and write it back to a file in a single function call.

Value

A rdocx, rpptx, or rxlsx object.

See Also

purrr::reduce(), vec_add_to_body()

Examples

```
## Not run:
if (interactive()) {
  x <- reduce_officer(value = LETTERS)

  officer_summary(x)
}

## End(Not run)
```

---

use_doc_version *Update the version of a document and optionally save a new version of the input document*

Description

[Experimental]

use_doc_version() is inspired by usethis::use_version() and designed to support semantic versioning for Microsoft Word or PowerPoint documents. The document version is tracked as a custom file property and a component of the document filename. If filename is supplied and the filename contains a version number, the function ignores any version number stored as a document property and overwrites the property with the new incremented version number.

doc_version() is a helper function that returns the document version based on a supplied filename or document properties.

The internals for this function are adapted from the internal idesc_bump_version() function authored by Csárdi Gábor for the {desc} package.

Usage

```
use_doc_version(
  filename = NULL,
  x = NULL,
  which = NULL,
  save = TRUE,
  sep = ".",
  property = "version",
  prefix = NULL,
  path = NULL,
  ...,
  call = caller_env()
)

doc_version(
  filename = NULL,
  x = NULL,
  sep = ".",
  property = "version",
  allow_new = TRUE,
  .default = c(0, 1, 0),
  call = caller_env()
)
```

Arguments

filename	Filename for a Word document or PowerPoint presentation. Optional if x is supplied, however, it is required to locally save a file with an updated version name.
x	A rdocx or rpptx object. Optional if filename is supplied.
which	A string specifying which level to increment, one of: "major", "minor", "patch", "dev".
save	If TRUE (default) and filename is supplied, write a new file replacing the existing version number.
sep	Character separating version components. Defaults to ".". "-" is also supported.
property	Property name optionally containing a version number. Defaults to "version".
prefix	Property name to use as prefix for new filename, e.g. "modified" to use modified date/time as the new filename prefix. If prefix is identical to property and the filename does not already include a version number, version is added to the filename as a prefix instead of a postfix. Defaults to NULL. Note handling of this parameter is expected to change in future versions of this function.
path	Passed to write_officer().
...	Additional parameters passed to write_officer() if save is TRUE.
call	The execution environment of a currently running function, e.g. caller_env(). The function will be mentioned in error messages as the source of the error. See the call argument of abort() for more information.

allow_new	If TRUE (default), return "0.1.0" if version can't be found in the filename or the properties of the input rdocx or rpptx object x.
.default	Specification for initial version.

Value

Invisibly return a rdocx or rpptx object with an updated version property.

---

vec_add_to_body *Add a vector of objects to a rdocx object using* add_to_body()

Description

vec_add_to_body() is a vectorized variant add_to_body() that allows users to supply a vector of values or str inputs to add multiple blocks of text, images, plots, or tables to a document. Alternatively, the function also supports adding a single object at multiple locations or using multiple different styles. All parameters are recycled using vctrs::vec_recycle_common() so inputs must be length 1 or match the length of the longest input vector. Optionally, the function can apply a separator between each element by passing a value to add_to_body() or passing docx to a function, such as officer::body_add_break().

Usage

```
vec_add_to_body(
  docx,
  ...,
  .sep = NULL,
  .pos = "after",
  .size = NULL,
  .call = caller_env()
)
```

Arguments

docx	A rdocx object.
...	Arguments passed on to add_to_body gt_object A gt object converted to an OOXML string with gt::as_word() then passed to add_xml_to_body() as str parameter. Required for add_gt_to_body(). keyword,id A keyword string used to place cursor with officer::cursor_reach() or bookmark id with officer::cursor_bookmark(). Defaults to NULL. If keyword or id are not provided, the gt object is inserted at the front of the document. index A integer matching a doc_index value appearing in a summary of the docx object created with officer::docx_summary(). If index is for a paragraph value, the text of the pargraph is used as a keyword.

call The execution environment of a currently running function, e.g. caller_env(). The function will be mentioned in error messages as the source of the error. See the call argument of abort() for more information.

value object to add in the document. Supported objects are vectors, data.frame, graphics, block of formatted paragraphs, unordered list of formatted paragraphs, pretty tables with package flextable, 'Microsoft' charts with package mschart.

style paragraph style name. These names are available with function styles_info and are the names of the Word styles defined in the base document (see argument path from read_docx).

str a wml string

pos where to add the new element relative to the cursor, one of "after", "before", "on".

.sep A bare function, such as officer::body_add_break or another object passed to add_to_body() as the value parameter.

.pos String passed to pos parameter if add_to_body() with .sep if .sep is not a function. Defaults to "after".

.size Desired output size.

.call The execution environment of a currently running function, e.g. caller_env(). The function will be mentioned in error messages as the source of the error. See the call argument of abort() for more information.

Examples

```
docx_example <- read_officer()

docx_example <- vec_add_to_body(
  docx_example,
  value = c("Sample text 1", "Sample text 2", "Sample text 3"),
  style = c("heading 1", "heading 2", "Normal")
)

docx_example <- vec_add_to_body(
  docx_example,
  value = rep("Text", 5),
  style = "Normal",
  .sep = officer::body_add_break
)

officer_summary(docx_example)

if (rlang::is_installed("gt")) {
  gt_tbl <- gt::gt(gt::gtcars[1:2, 1:2])

  # list inputs such as gt tables must be passed within a list to avoid
  # issues
  docx_example <- vec_add_to_body(
    docx_example,
    gt_object = list(gt_tbl, gt_tbl),
```

```
    keyword = c("Sample text 1", "Sample text 2")
  )

  officer_summary(docx_example)
}
```

`write_officer` *Write a rdocx, rpptx, or rxlsx object to a file*

Description

Write a rdocx, rpptx, or rxlsx object to a file

Usage

```
write_officer(x, path, overwrite = TRUE, modified_by = Sys.getenv("USER"), ...)
```

Arguments

`x`	A rdocx, rpptx, or rxlsx object to save. Document properties won't be set for rxlsx objects.
`path`	File path.
`overwrite`	If `TRUE` (default), remove file at path if it already exists. If `FALSE` and file exists, this function aborts.
`modified_by`	If the withr package is installed, modified_by overrides the default value for the lastModifiedBy property assigned to the output file by officer. Defaults to `Sys.getenv("USER")` (the same value used by officer).
`...`	Arguments passed on to `officer::set_doc_properties` `title,subject,creator,description` text fields `created` a date object `values` a named list (names are field names), each element is a single character value specifying value associated with the corresponding field name. If `values` is provided, argument `...` will be ignored.

Value

Returns the input object (invisibly) and writes the rdocx, rpptx, or rxlsx object to a file with a name and location matching the provided path.

Index