Package 'autoslider.core'

July 25, 2025

Type Package

Title Slide Automation for Tables, Listings and Figures

Version 0.2.7

Description The normal process of creating clinical study slides is that a statistician manually type in the numbers from outputs and a separate statistician to double check the typed in numbers. This process is time consuming, resource intensive, and error prone. Automatic slide generation is a solution to address these issues. It reduces the amount of work and the required time when creating slides, and reduces the risk of errors from manually typing or copying numbers from the output to slides. It also helps users to avoid unnecessary stress when creating large amounts of slide decks in a short time window.

License Apache License 2.0

URL https://github.com/insightsengineering/autoslider.core

BugReports https://github.com/insightsengineering/autoslider.core/issues

Depends R (>= 4.1.0)

Imports assertthat (>= 0.2.1), checkmate (>= 2.2.0), cli (>= 3.6.3), dplyr (>= 1.1.3), flextable (>= 0.9.6), forcats (>= 1.0.0), ggplot2 (>= 3.5.0), ggpubr (>= 0.6.0), grid, gridExtra (>= 2.3), gtsummary (>= 2.0.3), methods, officer (>= 0.6.7), rlang (>= 1.1.1), rlistings (>= 0.2.12), rtables (>= 0.6.13), rvg (>= 0.2.5), stringr (>= 1.5.0), survival, tern (>= 0.9.9), tidyr (>= 1.3.0), yaml (>= 2.3.7)

Suggests devtools (>= 2.4.5), ellmer (>= 0.2.0), filters (>= 0.3.1), formatters (>= 0.5.11), glue (>= 1.8.0), htmltools (>= 0.5.8.1), httr (>= 1.4.7), knitr (>= 1.49), lubridate (>= 1.9.3), mime (>= 0.12), nestcolor, purrr (>= 1.0.2), R.utils (>= 2.12.3), reticulate, rmarkdown (>= 2.23), rsvg (>= 0.3.4), styler (>= 1.10.2), svglite (>= 2.1.3), testthat (>= 3.2.0), TMB (>= 1.9.1), withr

VignetteBuilder knitr, rmarkdown

Config/Needs/verdepcheck insightsengineering/formatters, 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

LazyDataCompression xz

RoxygenNote 7.3.2

NeedsCompilation no

Author Joe Zhu [cre, aut] (ORCID: <https://orcid.org/0000-0001-7566-2787>), Heng Wang [aut], Yinqi Zhao [aut], Bo Ci [aut], Liming Li [aut], Laura Wang [ctb], Xiaoli Duan [aut], Stefan Pascal Thoma [aut], Thomas Neitmann [ctb], Miles Almond [aut], Mahdi About [ctb], Kai Lim [ctb], Nolan Steed [ctb], Daoling Pang [ctb], Elisabeth Deutschmann [ctb], Chenkai Lv [aut], Nina Qi [ctb], Jasmina Uzunovic [aut], Yolanda Zhou [aut]

Maintainer Joe Zhu <joe.zhu@roche.com>

Repository CRAN

Date/Publication 2025-07-25 10:20:09 UTC

Contents

`autoslider.core-package`

autoslider.core Package

Description

The normal process of creating clinical study slides is that a statistician manually type in the numbers from outputs and a separate statistician to double check the typed in numbers. This process is time consuming, resource intensive, and error prone. Automatic slide generation is a solution to address these issues. It reduces the amount of work and the required time when creating slides, and reduces the risk of errors from manually typing or copying numbers from the output to slides. It also helps users to avoid unnecessary stress when creating large amounts of slide decks in a short time window.

Author(s)

Maintainer: Joe Zhu <joe.zhu@roche.com> (ORCID)

Authors:

- Heng Wang
- Yinqi Zhao
- Bo Ci
- Liming Li
- Xiaoli Duan
- Stefan Pascal Thoma
- Miles Almond
- Chenkai Lv
- Jasmina Uzunovic
- Yolanda Zhou

Other contributors:

- Laura Wang [contributor]
- Thomas Neitmann [contributor]
- Mahdi About [contributor]
- Kai Lim [contributor]
- Nolan Steed [contributor]
- Daoling Pang [contributor]
- Elisabeth Deutschmann [contributor]
- Nina Qi [contributor]

See Also

Useful links:

- https://github.com/insightsengineering/autoslider.core
- Report bugs at https://github.com/insightsengineering/autoslider.core/issues

adding_ai_footnotes *Update footnote with AI response*

Description

Update footnote with AI response

Usage

```
adding_ai_footnotes(outputs, prompt_list, platform, base_url, api_key, model)
```

Arguments

outputs	Output objects
prompt_list	List of prompt
platform	Platform provider
base_url	Base url
api_key	API key
model	Model of choice

append_all_slides *Append All Predefined Slides to a PowerPoint Document*

Description

This function orchestrates the appending of a series of predefined slides (including title and section header slides) to a PowerPoint document based on a provided page list.

Usage

```
append_all_slides(
  doc_o,
  page_list = list(),
  doc_original = NULL,
  save_file = FALSE
)
```

Arguments

doc_o	Path to a PowerPoint (‘.pptx‘) file. Used to initialize the document if ‘doc_original‘ is ‘NULL‘, and for final post-processing.
page_list	A list of slide definitions. Each element in the list should be another list containing: - ‘type‘: A character string indicating the type of slide ("title" or "section"). - ‘to_page‘: An integer specifying the target page number for the slide. - Other arguments specific to the slide type (e.g., ‘study_id‘ for "title" slides, ‘section_title‘ for "section" slides).
doc_original	An optional existing ‘officer::rpptx‘ object. If ‘NULL‘, the document is initialized from ‘doc_o‘.
save_file	A logical value. If ‘TRUE‘, the final modified document is saved to a file after all slides have been appended.

Value

An ‘officer::rpptx‘ object with all specified slides appended.

Examples

```
tmp <- tempfile(fileext = ".pptx")
doc <- officer::read_pptx()
print(doc, target = tmp)

my_page_list <- list(
  list(type = "title", to_page = 1, study_id = "My Project"),
  list(type = "section", to_page = 2, section_title = "Introduction"),
  list(type = "title", to_page = 3, study_id = "Mid-Term Review"),
  list(type = "section", to_page = 4, section_title = "Key Findings")
)

# Append all slides using the dynamic page_list
doc <- append_all_slides(
  doc_o = tmp,
  page_list = my_page_list,
  save_file = TRUE
)
```

---

append_section_header_slides

Append Section Header Slides to a PowerPoint Document

Description

This function adds a new section header slide to an existing PowerPoint document using a "Section Header" layout from the "Office Theme". It populates the title placeholder with the provided section title.

Usage

```
append_section_header_slides(
  doc_o,
  section_title = "New Section",
  to_page = NA,
  doc_original = NULL,
  save_file = FALSE
)
```

Arguments

doc_o	Path to a PowerPoint (‘.pptx‘) file. Used to initialize the document if ‘doc_original‘ is ‘NULL‘, and for post-processing.
section_title	A character string for the title of the section header slide. Defaults to "New Section".
to_page	An integer specifying the page number where the new slide should be moved. If ‘NA‘, the slide is added at the end and ‘initialize_to_page‘ determines its final position.
doc_original	An optional existing ‘officer::rpptx‘ object. If ‘NULL‘, the document is initialized from ‘doc_o‘.
save_file	A logical value. If ‘TRUE‘, the modified document is saved to a file after adding the slide.

Value

An ‘officer::rpptx‘ object with the new section header slide appended.

Examples

```
tmp <- tempfile(fileext = ".pptx")
doc <- officer::read_pptx()
print(doc, target = tmp)

append_section_header_slides(
  doc_o = tmp,
  section_title = "My Section",
  to_page = 1,
  save_file = TRUE
)
```

---

append_title_slides *Append Title Slides to a PowerPoint Document*

Description

This function adds a new title slide using a "Title and Content" layout from the "Office Theme".

Usage

```
append_title_slides(
  doc_o,
  study_id = "XXXX change me",
  to_page = NA,
  doc_original = NULL,
  save_file = FALSE
)
```

Arguments

doc_o	Path to a PowerPoint (‘.pptx‘) file. Used to initialize the document if ‘doc_original‘ is ‘NULL‘.
study_id	A character string that represent your study identifier.
to_page	An integer specifying the page number where the new slide should be moved.
doc_original	An optional existing ‘officer::rpptx‘ object. If ‘NULL‘, the document is initialized from ‘doc_o‘.
save_file	A logical value. If ‘TRUE‘, the modified document is saved to a file after adding the slide.

Value

An ‘officer::rpptx‘ object with the new title slide appended.

Examples

```
tmp <- tempfile(fileext = ".pptx")
doc <- officer::read_pptx()
doc <- officer::add_slide(doc, layout = "Title Slide", master = "Office Theme")
print(doc, target = tmp)

doc <- append_title_slides(
  doc_o = tmp,
  study_id = "My Study #13",
  to_page = 1,
  save_file = TRUE
)
```

---

autoslider_error *autoslider_error class*

Description

autoslider_error class

Usage

```
autoslider_error(x, spec, step)
```

Arguments

x	character scaler
spec	spec should be a list containing "program" and "suffix"
step	step is a character indicating in which step the pipeline encounter error

Details

this function is used to create autoslider_error object. this function is for internal use only to create the autoslider_error object. It enable us for further functionalities, like providing help on easy debugging, e.g. if the error is inside the user function, provide the call and let the user run the code outside the pipeline.

Value

autoslider_error object

---

autoslider_format *Table color and font*

Description

Zebra themed color

Usage

```
autoslider_format(
  ft,
  odd_header = "#0EAED5",
  odd_body = "#EBF5FA",
  even_header = "#0EAED5",
  even_body = "#D0E4F2",
  font_name = "arial",
  body_font_size = 12,
  header_font_size = 14
)

blue_format(ft, ...)

orange_format(ft, ...)

red_format(ft, ...)

purple_format(ft, ...)

autoslider_dose_format(ft, header_vals = names(ft$body$dataset))

black_format_tb(ft, body_font_size = 8, header_font_size = 8, ...)
```

Arguments

ft	flextable object
odd_header	Hex color code, default to deep sky blue
odd_body	Hex color code, default to alice blue
even_header	Hex color code, default to slate gray
even_body	Hex color code, default to slate gray
font_name	Font name, default to arial
body_font_size	Font size of the table content, default to 12
header_font_size	Font size of the table header, default to 14
...	arguments passed to program
header_vals	Header

Value

A flextable with applied theme.

Functions

- autoslider_format(): User defined color code and font size
- blue_format(): Blue color theme
- orange_format(): Orange color theme
- red_format(): Red color theme
- purple_format(): Purple color theme
- autoslider_dose_format(): 'AutoslideR' dose formats
- black_format_tb(): Black color theme

Author(s)

Nina Qi and Jasmina Uzunovic

---

build_table_header *Build table header, a utility function to help with construct structured header for table layout*

Description

Build table header, a utility function to help with construct structured header for table layout

Usage

```
build_table_header(anl, arm, split_by_study, side_by_side)
```

Arguments

anl	analysis data object
arm	Arm variable for column split
split_by_study	if true, construct structured header with the study ID
side_by_side	A logical value indicating whether to display the data side by side.

Value

A ‘rtables‘ layout with desired header.

---

center_figure_loc *Create location container to center the figure, based on ppt size and user specified figure size*

Description

Create location container to center the figure, based on ppt size and user specified figure size

Usage

```
center_figure_loc(fig_width, fig_height, ppt_width, ppt_height)
```

Arguments

fig_width	Figure width
fig_height	Figure height
ppt_width	Slide width
ppt_height	Slide height

Value

Location for a placeholder from scratch

center_table_loc *create location container to center the table*

Description

create location container to center the table

Usage

```
center_table_loc(ft, ppt_width, ppt_height)
```

Arguments

ft	Flextable object
ppt_width	Powerpoint width
ppt_height	Powerpoint height

Value

Location for a placeholder

check_and_set_cutoff *Assert function to check the cutoff*

Description

Assert function to check the cutoff

Usage

```
check_and_set_cutoff(data, cutoff)
```

Arguments

data	dataframe
cutoff	cutoff threshold

Value

Set the cutoff value

`decorate` *generic function decorate*

Description

generic function decorate

s3 method for decorate

Usage

```
decorate(x, ...)

decorate(x, ...)
```

Arguments

`x`	object to decorate
`...`	additional arguments passed to methods

Value

No return value, called for side effects

`decorate,listing_df-method` *decorate listing*

Description

decorate listing

Usage

```
## S4 method for signature 'listing_df'
decorate(x, titles = "", footnotes = "", paper = "P8", for_test = FALSE, ...)
```

Arguments

`x`	A listing_df object representing the data to be decorated.
`titles`	Title to be added to the table.
`footnotes`	Footnote to be added to the table
`paper`	Orientation and font size as string, e.g. "P8"; "L11"
`for_test`	‘logic‘ CICD parameter
`...`	Additional arguments. not used.

Value

No return value, called for side effects

`decorate,VTableTree-method`
Decorate TableTree

Description

Decorate TableTree

Usage

```
## S4 method for signature 'VTableTree'
decorate(x, titles = "", footnotes = "", paper = "P8", for_test = FALSE, ...)
```

Arguments

`x`	A VTableTree object representing the data to be decorated.
`titles`	Title to be added to the table.
`footnotes`	Footnote to be added to the table
`paper`	Orientation and font size as string, e.g. "P8"; "L11"
`for_test`	‘logic‘ CICD parameter
`...`	Additional arguments passed to the decoration function.

Value

No return value, called for side effects

`decorate.autoslider_error`
decorate method for autoslider_error class

Description

decorate method for autoslider_error class

Usage

```
decorate.autoslider_error(x, ...)
```

Arguments

`x`	object to decorate
`...`	additional arguments. not used.

Value

No return value, called for side effects

`decorate.default` *default method to decorate*

Description

default method to decorate

Usage

```
decorate.default(x, ...)
```

Arguments

`x`	object to decorate
`...`	additional arguments. not used.

Value

No return value, called for side effects

`decorate.ggplot` *Decorate ggplot object*

Description

Decorate ggplot object

Usage

```
decorate.ggplot(
  x,
  titles = "",
  footnotes = "",
  paper = "L11",
  for_test = FALSE,
  ...
)
```

Arguments

x	An object to decorate
titles	Plot titles
footnotes	Plot footnotes
paper	Paper size, by default "L11"
for_test	'logic' CICD parameter
...	additional arguments. not used.

Details

The paper default paper size, 'L11', indicate that the fontsize is 11. The fontsize of the footnotes, is the fontsize of the titles minus 2.

Value

No return value, called for side effects

---

decorate.grob *decorate grob*

Description

decorate grob

Usage

```
decorate.grob(
  x,
  titles = "",
  footnotes = "",
  paper = "L11",
  for_test = FALSE,
  ...
)
```

Arguments

x	object to decorate
titles	graph titles
footnotes	graph footnotes
paper	paper size. default is "L8".
for_test	'logic' CICD parameter
...	Additional arguments. not used.

Details

The paper default paper size, ‘L11‘, indicate that the fontsize is 11. The fontsize of the footnotes, is the fontsize of the titles minus 2.

Value

No return value, called for side effects

`decorate.gtsummary` *decorate gtsummary*

Description

decorate gtsummary

Usage

```
decorate.gtsummary(
  x,
  titles = "",
  footnotes = "",
  paper = "L11",
  for_test = FALSE,
  ...
)
```

Arguments

`x`	gtsummary object to decorate
`titles`	graph titles
`footnotes`	graph footnotes
`paper`	paper size. default is "L8".
`for_test`	‘logic‘ CICD parameter
`...`	Additional arguments. not used.

Details

The paper default paper size, ‘L11‘, indicate that the fontsize is 11. The fontsize of the footnotes, is the fontsize of the titles minus 2.#’

Value

No return value, called for side effects

decorate.list *decorate list of grobs*

Description

decorate list of grobs

Usage

```
decorate.list(x, titles, footnotes, paper = "L11", for_test = FALSE, ...)
```

Arguments

x	object to decorate
titles	graph titles
footnotes	graph footnotes
paper	paper size. default is "L11".
for_test	'logic' CICD parameter
...	additional arguments. not used

Details

The paper default paper size, 'L11', indicate that the fontsize is 11. The fontsize of the footnotes, is the fontsize of the titles minus 2.

Value

No return value, called for side effects

decorate_outputs *Decorate outputs*

Description

Decorate outputs with titles and footnotes

Usage

```
decorate_outputs(
  outputs,
  generic_title = NULL,
  generic_footnote = "Confidential and for internal use only",
  version_label = get_version_label_output(),
  for_test = FALSE
)
```

Arguments

outputs	‘list‘ of output objects as created by ‘generate_outputs‘
generic_title	‘character‘ vector of titles
generic_footnote	‘character‘ vector of footnotes
version_label	‘character‘. A version label to be added to the title.
for_test	‘logic‘ CICD parameter

Details

‘generic_title‘ and ‘generic_footnote‘ will be added to *all* outputs. The use case is to add information such as protocol number and snapshot date defined in a central place (e.g. metadata.yml) to *every* output.

‘version_label‘ must be either ‘"DRAFT"‘, ‘"APPROVED"‘ or ‘NULL‘. By default, when outputs are created on the master branch it is set to ‘NULL‘, i.e. no version label will be displayed. Otherwise ‘"DRAFT"‘ will be added. To add ‘"APPROVED"‘ to the title you will need to explicitly set ‘version_label = "APPROVED"‘.

Value

No return value, called for side effects

---

dec_paste *Concatenate arguments into a string*

Description

Concatenate arguments into a string

Usage

```
dec_paste(...)
```

Arguments

...	arguments passed to program

Value

No return value, called for side effects

eg_adae *Cached ADAE*

Description

Cached ADAE data

Usage

```
data(eg_adae)
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 1934 rows and 93 columns.

eg_adeg *Cached ADEG*

Description

Cached ADEG data

Usage

```
data(eg_adeg)
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 13600 rows and 88 columns.

eg_adex *Cached ADEX*

Description

Cached ADEX data

Usage

```
data(eg_adex)
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 6400 rows and 79 columns.

eg_adlb *Cached ADLB*

Description

Cached ADLB data

Usage

```
data(eg_adlb)
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 8400 rows and 102 columns.

eg_adrs *Cached ADRS*

Description

Cached ADRS data

Usage

```
data(eg_adrs)
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 3200 rows and 65 columns.

eg_adsl *Cached ADSL*

Description

Cached ADSL data

Usage

```
data(eg_adsl)
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 400 rows and 56 columns.

eg_adtr *Cached ADTR*

Description

Cached ADTR data

Usage

```
data(eg_adtr)
```

Format

An object of class `data.frame` with 2800 rows and 76 columns.

eg_adtte *Cached ADTTE*

Description

Cached ADTTE data

Usage

```
data(eg_adtte)
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 2000 rows and 67 columns.

eg_advs *Cached ADVS*

Description

Cached ADVS data

Usage

```
data(eg_advs)
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 16800 rows and 87 columns.

fastDoCall *Does do.call quicker, and avoids issues with debug mode within do.call*

Description

copied from ms showcase app

Usage

```
fastDoCall(what, args, quote = FALSE, envir = parent.frame())
```

Arguments

what	either a function or a non-empty character string naming the function to be called.
args	a list of arguments to the function call. The names attribute of args gives the argument names.
quote	a logical value indicating whether to quote the arguments.
envir	an environment within which to evaluate the call. This will be most useful if what is a character string and the arguments are symbols or quoted expressions.

Value

No return value, called for side effects

figure_to_slide *Add figure to slides*

Description

Add figure to slides

Usage

```
figure_to_slide(
  ppt,
  content,
  decor = TRUE,
  fig_width,
  fig_height,
  figure_loc = ph_location_type("body"),
  fig_editable = FALSE,
  ...
)
```

Arguments

ppt	slide page
content	content to be added
decor	should decoration be added
fig_width	user specified figure width
fig_height	user specified figure height
figure_loc	location of the figure. Defaults to ‘ph_location_type("body")‘
fig_editable	whether we want the figure to be editable in pptx viewers
...	arguments passed to program

Value

slide with the added content

---

filter_spec *Filter a spec object*

Description

Filter a spec object

Usage

```
filter_spec(spec, filter_expr, verbose = TRUE)
```

Arguments

spec	A ‘spec‘ object as returned by ‘read_spec()‘
filter_expr	A ‘logical‘ expression indicating outputs to keep
verbose	Should a message about the number of outputs matching ‘filter_spec‘ be printed? Defaults to ‘TRUE‘.

Value

A ‘spec‘ object containing only the outputs matching ‘filter_expr‘

Author(s)

Thomas Neitmann (‘neitmant‘)

Examples

```
library(dplyr)
spec_file <- system.file("spec.yml", package = "autoslider.core")
spec <- spec_file %>% read_spec()

## Keep only the t_dm_IT output
filter_spec(spec, output == "t_dm_IT")

## Same as above but more verbose
filter_spec(spec, program == "t_dm" && suffix == "IT")

## Keep all t_ae outputs
filter_spec(spec, program == "t_ae")

## Keep all output run on safety population
filter_spec(spec, "SE" %in% suffix)

## Keep t_dm_CHN_IT and t_dm_CHN_SE
filter_spec(spec, program == "t_dm" && suffix %in% c("CHN_IT", "CHN_SE"))

## Keep all tables
filter_spec(spec, grepl("^t_", program))
```

`format_3d` *Format of xx.xx (xx.xx, xx.xx)*

Description

Format of xx.xx (xx.xx, xx.xx)

Usage

```
format_3d(x, output)
```

Arguments

`x`	input array
`output`	output handle

Value

formatted values

format_date *Convert dates from 'yyyy-mm-dd' format into 20APR2019 format 'Datetime' format removes the time and outputs date in the same way Able to handle truncated dates as well (e.g. just the year or year and month)*

Description

'dplyr::case_when()' will check all RHS expressions on the input, this means if these expressions return warnings, they will happen even then the input doesn't doesn't satisfy the LHS. For this reason, I had to 'quiet' all 'lubridate' functions. This 'format_date()' function was tested with the inputs in the examples, all gave the expected returned value, so there should be no issues.

Usage

```
format_date(x)
```

Arguments

x	vector of dates in character, in 'yyyy-mm-dd' format

Value

A vector.

Examples

```
require(lubridate)

# expected to return "2019"
format_date("2019")

# expected to return "20APR2019"
format_date("2019-04-20")

# expected to return ""
format_date("")

# expected to return "18JUN2019"
format_date("2019-06-18T10:32")

# expected to return "APR2019"
format_date("2019-04")
```

func_wrapper *function wrapper to pass filtered data*

Description

function wrapper to pass filtered data

Usage

```
func_wrapper(func, datasets, spec, verbose = TRUE)
```

Arguments

func	function name
datasets	list of raw datasets
spec	spec
verbose	whether to show verbose information

Value

a wrapped function using filtered adam

generate_output *Generate output and apply filters, titles, and footnotes*

Description

Generate output and apply filters, titles, and footnotes

Usage

```
generate_output(program, datasets, spec, verbose_level = 2, ...)
```

Arguments

program	program name
datasets	list of datasets
spec	spec
verbose_level	Verbose level of messages be displayed. See details for further information.
...	arguments passed to program

Details

'verbose_level' is used to control how many messages are printed out. By default, '2' will show all filter messages and show output generation message. '1' will show output generation message only. '0' will display no message.

Value

No return value, called for side effects

Author(s)

Liming Li ('Lil128')

Examples

```
library(dplyr)
filters::load_filters(
  yaml_file = system.file("filters.yml", package = "autoslider.core"),
  overwrite = TRUE
)

spec_file <- system.file("spec.yml", package = "autoslider.core")
spec <- spec_file %>% read_spec()

data <- list(
  adsl = eg_adsl,
  adae = eg_adae
)
generate_output("t_ae_slide", data, spec$t_ae_slide_SE)
```

`generate_outputs` *Generate all outputs from a spec*

Description

Generate all outputs from a spec

Usage

```
generate_outputs(spec, datasets, verbose_level = 2)
```

Arguments

`spec`	Specification list generated by 'read_spec'
`datasets`	A 'list' of datasets
`verbose_level`	Verbose level of messages be displayed. See details for further information.

Details

'verbose_level' is used to control how many messages are printed out. By default, '2' will show all filter messages and show output generation message. '1' will show output generation message only. '0' will display no message.

Value

No return value, called for side effects

Author(s)

- Thomas Neitmann ('neitmant') - Liming Li ('Lil128')

Examples

```
library(dplyr, warn.conflicts = FALSE)
data <- list(
  adsl = eg_adsl,
  adae = eg_adae
)
filters::load_filters(
  yaml_file = system.file("filters.yml", package = "autoslider.core"),
  overwrite = TRUE
)

spec_file <- system.file("spec.yml", package = "autoslider.core")
spec_file %>%
  read_spec() %>%
  filter_spec(output %in% c("t_dm_slide_IT", "t_ae_slide_SE")) %>%
  generate_outputs(datasets = data)
```

---

generate_slides *generate slides based on output*

Description

generate slides based on output

Usage

```
generate_slides(
  outputs,
  outfile = paste0(tempdir(), "/output.pptx"),
 template = file.path(system.file(package = "autoslider.core"), "theme/basic.pptx"),
  fig_width = 9,
  fig_height = 5,
  t_lpp = 20,
```

```
  t_cpp = 200,
  l_lpp = 20,
  l_cpp = 150,
  fig_editable = FALSE,
  ...
)
```

Arguments

outputs	List of output
outfile	Out file path
template	Template file path
fig_width	figure width in inch
fig_height	figure height in inch
t_lpp	An integer specifying the table lines per page Specify this optional argument to modify the length of all of the table displays
t_cpp	An integer specifying the table columns per page Specify this optional argument to modify the width of all of the table displays
l_lpp	An integer specifying the listing lines per page Specify this optional argument to modify the length of all of the listings display
l_cpp	An integer specifying the listing columns per page Specify this optional argument to modify the width of all of the listings display
fig_editable	whether we want the figure to be editable in pptx viewers, defaults to FALSE
...	arguments passed to program

Value

No return value, called for side effects

Examples

```
# Example 1. When applying to the whole pipeline
library(dplyr)
data <- list(
  adsl = eg_adsl %>% dplyr::mutate(FASFL = SAFFL),
  adae = eg_adae
)


filters::load_filters(
  yaml_file = system.file("filters.yml", package = "autoslider.core"),
  overwrite = TRUE
)


spec_file <- system.file("spec.yml", package = "autoslider.core")
```

```
spec_file %>%
  read_spec() %>%
  filter_spec(program %in% c("t_dm_slide")) %>%
  generate_outputs(datasets = data) %>%
  decorate_outputs() %>%
  generate_slides()

# Example 2. When applying to an rtable object or an rlisting object
adsl <- eg_adsl
t_dm_slide(adsl, "TRT01P", c("SEX", "AGE")) %>%
  generate_slides()
```

---

gen_notes *General notes*

Description

General notes

Usage

```
gen_notes()
```

Note

* Default arm variables are set to ‘"TRT01A"‘ for safety output, and ‘"TRT01P"‘ for efficacy output

---

get_ellmer_chat *Get an ‘ellmer‘ chat API with given platform*

Description

Get an ‘ellmer‘ chat API with given platform

Usage

```
get_ellmer_chat(
  platform = "deepseek",
  base_url = "https://api.deepseek.com",
  api_key = get_deepseek_key(),
  model = "deepseek-chat"
)
```

Arguments

platform	Platform provider
base_url	Base url
api_key	API key
model	Model of choice

---

get_prompt_list *Read prompt list from yaml file*

Description

Read prompt list from yaml file

Usage

```
get_prompt_list(filename)
```

Arguments

filename	File name

---

get_proper_title *Adjust title line break and font size*

Description

Adjust title line break and font size

Usage

```
get_proper_title(title, max_char = 60, title_color = "#1C2B39")
```

Arguments

title	Character string
max_char	Integer specifying the maximum number of characters in one line
title_color	Title color

gt_t_dm_slide *Demographic table with gtsummary*

Description

Demographic table with gtsummary

Usage

```
gt_t_dm_slide(adsl, arm = "TRT01P", vars = c("AGE", "SEX", "RACE"))
```

Arguments

adsl	ADSL data set, dataframe
arm	Arm variable, character, "‘TRT01P" by default.
vars	Characters of variables

Value

gtsummary object

Note

* Default arm variables are set to ‘"TRT01A"‘ for safety output, and ‘"TRT01P"‘ for efficacy output

Examples

```
library(dplyr)
adsl <- eg_adsl
out1 <- gt_t_dm_slide(adsl, "TRT01P", c("SEX", "AGE", "RACE", "ETHNIC", "COUNTRY"))
print(out1)
generate_slides(out1, paste0(tempdir(), "/dm.pptx"))
```

g_eg_slide *Plot mean values of EG*

Description

Wrapper for ‘g_mean_general()‘. Requires filtering of the datasets (e.g. using SUFFIX in spec.yml)

Usage

```
g_eg_slide(
  adsl,
  adeg,
  arm = "TRT01P",
  paramcd = "PARAM",
  subtitle = "Plot of Mean and 95% Confidence Limits by Visit.",
  ...
)
```

Arguments

adsl	ADSL data
adeg	ADVS data
arm	‘"TRT01P"‘ by default
paramcd	Which variable to use for plotting. By default ‘"PARAM"‘
subtitle	character scalar forwarded to g_lineplot
...	| Gets forwarded to ‘tern::g_lineplot()‘. This lets you specify additional arguments to ‘tern::g_lineplot()‘

Author(s)

Stefan Thoma (‘thomas7‘)

Examples

```
library(dplyr)

adeg_filtered <- eg_adeg %>% filter(
  PARAMCD == "HR"
)
plot_eg <- g_eg_slide(
  adsl = eg_adsl,
  adeg = adeg_filtered,
  arm = "TRT01P",
  paramcd = "PARAM",
  subtitle_add_unit = FALSE
) +
  ggplot2::theme(axis.text.x = ggplot2::element_text(angle = 45, hjust = 1))

generate_slides(plot_eg, paste0(tempdir(), "/g_eg.pptx"))
```

g_lb_slide *Plot mean values of LB*

Description

Wrapper for ‘g_mean_general()‘. Requires filtering of the datasets (e.g. using SUFFIX in spec.yml)

Usage

```
g_lb_slide(
  adsl,
  adlb,
  arm = "TRT01P",
  paramcd = "PARAM",
  y = "AVAL",
  subtitle = "Plot of Mean and 95% Confidence Limits by Visit.",
  ...
)
```

Arguments

adsl	ADSL data
adlb	ADLB data
arm	‘"TRT01P"‘ by default
paramcd	character scalar. defaults to By default ‘"PARAM"‘ Which variable to use for plotting.
y	character scalar. Variable to plot on the Y axis. By default ‘"AVAL"‘
subtitle	character scalar forwarded to g_lineplot
...	| Gets forwarded to ‘tern::g_lineplot()‘. This lets you specify additional arguments to ‘tern::g_lineplot()‘

Author(s)

Stefan Thoma (‘thomas7‘)

Examples

```
library(dplyr)

adlb_filtered <- eg_adlb %>% filter(
  PARAMCD == "CRP"
)
plot_lb <- g_lb_slide(
  adsl = eg_adsl,
  adlb = adlb_filtered,
  paramcd = "PARAM",
```

```
  subtitle_add_unit = FALSE
) +
  ggplot2::theme(axis.text.x = ggplot2::element_text(angle = 45, hjust = 1))
generate_slides(plot_lb, paste0(tempdir(), "/g_lb.pptx"))

# Let's plot change values:
plot_lb_chg <- g_lb_slide(
  adsl = eg_adsl,
  adlb = adlb_filtered,
  paramcd = "PARAM",
  y = "CHG",
  subtitle = "Plot of change from baseline and 95% Confidence Limit by Visit."
)
generate_slides(plot_lb_chg, paste0(tempdir(), "/g_lb_chg.pptx"))
```

`g_mean_general` *Plot mean values general function used by wrappers 'g_vs_slide','g_lb_slide', & 'g_eg_slide'*

Description

adapted from https://insightsengineering.github.io/tlg-catalog/stable/graphs/other/mng01.html

Usage

```
g_mean_general(
  adsl,
  data,
  variables = control_lineplot_vars(group_var = "TRT01P"),
  by_vars = c("USUBJID", "STUDYID"),
  subtitle = "Plot of Mean and 95% Confidence Limits by Visit.",
  ...
)
```

Arguments

`adsl`	ADSL dataset
`data`	dataset containing the variable of interest in PARAMCD and AVAL
`variables`	(named `character`) vector of variable names in `df` which should include: • `x` (`string`) name of x-axis variable. • `y` (`string`) name of y-axis variable. • `group_var` (`string` or `NULL`) name of grouping variable (or strata), i.e. treatment arm. Can be `NA` to indicate lack of groups.

- subject_var (string or NULL)
  name of subject variable. Only applies if group_var is not NULL.
- paramcd (string or NA)
  name of the variable for parameter's code. Used for y-axis label and plot's subtitle. Can be NA if paramcd is not to be added to the y-axis label or subtitle.
- y_unit (string or NA)
  name of variable with units of y. Used for y-axis label and plot's subtitle. Can be NA if y unit is not to be added to the y-axis label or subtitle.
- facet_var (string or NA)
  name of the secondary grouping variable used for plot faceting, i.e. treatment arm. Can be NA to indicate lack of groups.

by_vars	variables to merge the two datasets by
subtitle	character scalar forwarded to g_lineplot
...	additional arguments passed to 'tern::g_lineplot'

Author(s)

Stefan Thoma ('thomas7')

Examples

```
library(dplyr)
advs_filtered <- eg_advs %>% filter(
  PARAMCD == "SYSBP"
)
out1 <- g_mean_general(eg_adsl, advs_filtered)
generate_slides(out1, paste0(tempdir(), "/g_mean.pptx"))
```

---

g_vs_slide *Plot mean values of VS*

Description

Wrapper for 'g_mean_general()'. Requires filtering of the datasets (e.g. using SUFFIX in spec.yml)

Usage

```
g_vs_slide(
  adsl,
  advs,
  arm = "TRT01P",
  paramcd = "PARAM",
  subtitle = "Plot of Mean and 95% Confidence Limits by Visit.",
  ...
)
```

Arguments

adsl	ADSL data
advs	ADVS data
arm	'"TRT01P"' by default
paramcd	Which variable to use for plotting. By default '"PARAM"'
subtitle	character scalar forwarded to g_lineplot
...	| Gets forwarded to 'tern::g_lineplot()'. This lets you specify additional arguments to 'tern::g_lineplot()'

Author(s)

Stefan Thoma ('thomas7')

Examples

```
library(dplyr)
advs_filtered <- eg_advs %>% filter(
  PARAMCD == "SYSBP"
)
plot_vs <- g_vs_slide(
  adsl = eg_adsl,
  advs = advs_filtered,
  paramcd = "PARAM",
  subtitle_add_unit = FALSE
) +
  ggplot2::theme(axis.text.x = ggplot2::element_text(angle = 45, hjust = 1))
# makes editable plots
generate_slides(plot_vs, paste0(tempdir(), "/g_vs.pptx"), fig_editable = TRUE)
# not editable plots, which appear as images
generate_slides(plot_vs, paste0(tempdir(), "/g_vs.pptx"), fig_editable = FALSE)
```

---

initialize_doc_original

Initialize PowerPoint Document Object

Description

This function ensures a PowerPoint document ('officer::rpptx' object) is loaded. If a 'doc_original' is provided, it is directly returned. Otherwise, the function reads the presentation from the given file path.

Usage

```
initialize_doc_original(doc_original, doc_o)
```

Arguments

doc_original	An existing ‘officer::rpptx‘ object, or ‘NULL‘ to read from file.
doc_o	Path to a PowerPoint (‘.pptx‘) file. Used only if ‘doc_original‘ is ‘NULL‘.

Value

An ‘officer::rpptx‘ PowerPoint object.

Examples

```
example <- tempfile(fileext = ".pptx")
doc <- officer::read_pptx()
doc <- officer::add_slide(doc, layout = "Title and Content")
print(doc, target = example)
doc <- initialize_doc_original(NULL, example)
```

---

initialize_to_page *Determine Slide Insertion Page Number*

Description

Computes the appropriate page number at which to insert a new slide into the PowerPoint deck. Defaults to appending to the end if no value is provided.

Usage

```
initialize_to_page(doc_original, to_page)
```

Arguments

doc_original	An ‘officer::rpptx‘ object representing the PowerPoint file.
to_page	Desired slide index to insert the new slide. If ‘NA‘, appends to the last page.

Value

A single integer value indicating the validated page number for slide insertion.

Examples

```
tmp <- tempfile(fileext = ".pptx")
doc <- officer::read_pptx()
doc <- officer::add_slide(doc, layout = "Title Slide", master = "Office Theme")
print(doc, target = tmp)
doc <- officer::read_pptx(tmp)
initialize_to_page(doc, NA) # append to end
initialize_to_page(doc, 1) # insert at page 1
```

list_all_templates *[EXPERIMENTAL] List All Available Templates*

Description

[EXPERIMENTAL] List All Available Templates

Usage

```
list_all_templates(package = "autoslider.core")
```

Arguments

package	Which package to search for the template files. Defaults to "autoslider.core".

Value

A character vector of available template names in the specified package.

Examples

```
list_all_templates(package = "autoslider.core")
```

lyt_to_side_by_side *Build side by side layout by cbind*

Description

Build side by side layout by cbind

Usage

```
lyt_to_side_by_side(lyt, anl, side_by_side = NULL)
```

Arguments

lyt	layout object
anl	analysis data object
side_by_side	A logical value indicating whether to display the data side by side.

Value

An ‘rtables‘ layout

lyt_to_side_by_side_two_data *Build side by side layout by cbind*

Description

Build side by side layout by cbind

Usage

```
lyt_to_side_by_side_two_data(lyt, anl, alt_counts_df, side_by_side = NULL)
```

Arguments

lyt	layout object
anl	analysis data object
alt_counts_df	alternative data frame for counts
side_by_side	A logical value indicating whether to display the data side by side.

Value

An ‘rtables‘ layout

l_ae_slide *Adverse Events listing adapted from https://insightsengineering.github.io/tlg-catalog/stable/listings/adverse-events/ael02.html*

Description

Adverse Events listing adapted from https://insightsengineering.github.io/tlg-catalog/stable/listings/adverse-events/ael02.html

Usage

```
l_ae_slide(adsl, adae)
```

Arguments

adsl	ADSL data
adae	ADAE data

Examples

```
library(dplyr)
library(rlistings)
adsl <- eg_adsl
adae <- eg_adae

out <- l_ae_slide(adsl = adsl, adae = adae)
head(out)
```

`mutate_actarm` *Refactor active arm*

Description

Refactor active arm

Usage

```
mutate_actarm(
  df,
  arm_var = "TRT01A",
  levels = c("PLACEBO + PACLITAXEL + CISPLATIN",
    "ATEZOLIZUMAB + TIRAGOLUMAB + PACLITAXEL + CISPLATIN"),
  labels = c("Pbo+Pbo+PC", "Tira+Atezo+PC")
)
```

Arguments

`df`	Input dataframe
`arm_var`	Arm variable
`levels`	factor levels
`labels`	factor labels

Value

Dataframe with re-level and re-labelled arm variable.

na_replace *Replace NAs to NA*

Description

Replace NAs to NA

Usage

```
na_replace(table_df)
```

Arguments

table_df	Table dataframe

Value

Input dataframe with both column replaced to NA

new_round *Founding method*

Description

Founding method

Usage

```
new_round(x, digits = 1)
```

Arguments

x	number need to be rounded
digits	number of digits

Value

rounded value

null_report *Null report*

Description

Null report

Usage

```
null_report()
```

Details

This will create a null report similar as STREAM does. You can use it inside output functions as shown in the example below.

Value

An empty ‘rtables‘ object

Author(s)

Thomas Neitmann (‘neitmant‘)

Examples

```
library(dplyr)
library(filters)
data <- list(
  adsl = eg_adsl,
  adae = eg_adae %>% mutate(AREL = "")
)

null_report()

## An example how to use the `null_report()` inside an output function
t_ae <- function(datasets) {
  trt <- "ACTARM"
  anl <- semi_join(
    datasets$adae,
    datasets$adsl,
    by = c("STUDYID", "USUBJID")
  )

  return(null_report())
}

data %>%
  filters::apply_filter("SER_SE") %>%
```

```
t_ae()
```

`perc_perc` *Format of (xx%, xx%)*

Description

Format of (xx%, xx%)

Usage

```
perc_perc(x, output)
```

Arguments

`x`	input array
`output`	output handle

Value

formatted values

`ph_with_img` *Placeholder for ph_with_img*

Description

Placeholder for ph_with_img

Usage

```
ph_with_img(ppt, figure, fig_width, fig_height, figure_loc)
```

Arguments

`ppt`	power point file
`figure`	image object
`fig_width`	width of figure
`fig_height`	height of figure
`figure_loc`	location of figure

Value

Location for a placeholder

postprocessing_doc *Post-process PowerPoint Document*

Description

Performs final actions on the PowerPoint object, including optionally saving the updated file. The saved filename includes a suffix indicating the slide type that was appended.

Usage

```
postprocessing_doc(doc, save_file, doc_o, type = "")
```

Arguments

doc	An ‘officer::rpptx‘ object to finalize.
save_file	A boolean indicating whether to save the file to disk.
doc_o	Original PowerPoint file path.
type	A string suffix to label the output file, e.g., ‘"cohort_sec"‘ or ‘"safety_sum_sec"‘.

Value

The modified ‘officer::rpptx‘ object.

Examples

```
tmp <- tempfile(fileext = ".pptx")
doc <- officer::read_pptx()
doc <- officer::add_slide(doc, layout = "Title Slide", master = "Office Theme")
print(doc, target = tmp)
doc <- officer::read_pptx(tmp)
# Call postprocessing_doc to save a modified version of doc
postprocessing_doc(doc, TRUE, tmp, type = "final")
```

preprocess_t_dd *Preprocess t_dd function*

Description

Preprocess t_dd function

Usage

```
preprocess_t_dd(
  df,
  levels = c("PROGRESSIVE DISEASE", "ADVERSE EVENT", "OTHER", "<Missing>"),
  labels = c("Progressive Disease", "Adverse Events", "Other", "<Missing>")
)
```

Arguments

df	Input dataframe
levels	factor levels
labels	factor labels

Value

dataframe

---

preprocess_t_ds *Preprocess t_ds function*

Description

Preprocess t_ds function

Usage

```
preprocess_t_ds(
  df,
  levels = c("Alive: On Treatment", "Alive: In Follow-up", "<Missing>"),
  labels = c("Alive: On Treatment", "Alive: In Follow-up", "<Missing>")
)
```

Arguments

df	Input dataframe
levels	factor levels
labels	factor labels

Value

dataframe

print.decoratedGrob *Print decorated grob*

Description

Print decorated grob

Usage

```
## S3 method for class 'decoratedGrob'
print(x, ...)
```

Arguments

x	An object of class ‘decoratedGrob‘
...	not used.

Value

No return value, called for side effects

print.decoratedGrobSet *Print decorated grob set*

Description

Print decorated grob set

Usage

```
## S3 method for class 'decoratedGrobSet'
print(x, ...)
```

Arguments

x	An object of class ‘decoratedGrobSet‘
...	not used.

Value

No return value, called for side effects

read_spec *Read yaml spec file*

Description

Read yaml spec file and split according to filter lists

Usage

```
read_spec(spec_file = "spec.yml", metadata = NULL)
```

Arguments

spec_file	'character'. Path to a yaml spec file
metadata	Metadata of study

Value

An object of class 'spec' which is a 'list' where each element corresponds to one output, e.g. 't_dm_IT'.

Author(s)

- Liming Li ('Lil128') - Thomas Neitmann ('neitmant') - Joe Zhu

Examples

```
spec_file <- system.file("spec.yml", package = "autoslider.core")

## Take a look at the 'raw' content of the spec file
cat(readLines(spec_file)[1:24], sep = "\n")

## This is how it looks once read into R
spec <- read_spec(spec_file)
spec[1:3]
```

save_output *Save an Output*

Description

Save an Output

Usage

```
save_output(output, file_name, save_rds = TRUE)

save_output(output, file_name, save_rds = TRUE)

save_output.autoslider_error(output, file_name, save_rds = TRUE)

## S4 method for signature 'dVTableTree'
save_output(output, file_name, save_rds = TRUE)

save_output.decoratedGrob(output, file_name, save_rds = TRUE)

save_output.decoratedGrobSet(output, file_name, save_rds = TRUE)

save_output.dgtsummary(output, file_name, save_rds = TRUE)

save_output.dlisting(output, file_name, save_rds = TRUE)
```

Arguments

output	Output object, e.g. an ‘rtable‘ or ‘grob‘
file_name	Full path of the new file *excluding* the extension
save_rds	Saved as an ‘.rds‘ files

Details

Tables are saved as RDS file

Value

The input ‘object‘ invisibly

No return value, called for side effects

The input ‘object‘ invisibly

The input ‘object‘ invisibly

The input ‘object‘ invisibly

The input ‘object‘ invisibly

Examples

```
library(dplyr)
adsl <- eg_adsl %>%
  filter(SAFFL == "Y") %>%
  mutate(TRT01P = factor(TRT01P, levels = c("A: Drug X", "B: Placebo")))
output_dir <- tempdir()
t_dm_slide(adsl, "TRT01P", c("SEX", "AGE", "RACE", "ETHNIC", "COUNTRY")) %>%
  decorate(
    title = "Demographic table",
```

```
  footnote = ""
) %>%
save_output(
  file_name = file.path(output_dir, "t_dm_SE"),
  save_rds = TRUE
)
```

---

save_outputs *Save a list of outputs*

Description

Save a list of outputs

Usage

```
save_outputs(
  outputs,
  outfolder = file.path("output"),
  generic_suffix = NULL,
  save_rds = TRUE,
  verbose_level = 1
)
```

Arguments

outputs	'list' of outputs as created by 'generate_outputs'
outfolder	Folder in which to store the 'outputs"
generic_suffix	generic suffix. must be length 1 character or NULL.
save_rds	Should the input 'outputs' be saved as '.rds' files in in addition to '.out' or '.pdf' files? Defaults to 'FALSE'.
verbose_level	Level of verbose information displayed. Default set to '1'.

Value

The input 'object' invisibly

Examples

```
## As `save_outputs` is the last step in the pipeline we have to run
## the 'whole machinery' in order to show its functionality.
library(dplyr, warn.conflicts = FALSE)

data <- list(
  adsl = eg_adsl,
  adae = eg_adae,
```

```
  adtte = eg_adtte
)

filters::load_filters(
  yaml_file = system.file("filters.yml", package = "autoslider.core"),
  overwrite = TRUE
)

## For this example the outputs will be saved in a temporary directory. In a
## production run this should be the reporting event's 'output' folder instead.
output_dir <- tempdir()

spec_file <- system.file("spec.yml", package = "autoslider.core")
read_spec(spec_file) %>%
  filter_spec(program == "t_dm_slide") %>%
  generate_outputs(datasets = data) %>%
  decorate_outputs() %>%
  save_outputs(outfolder = output_dir)
```

`slides_from_rds` *Generate slides from rds files*

Description

Generate slides from rds files

Usage

```
slides_from_rds(
  filenames,
  outfile = paste0(tempdir(), "/output.pptx"),
 template = file.path(system.file(package = "autoslider.core"), "theme/basic.pptx")
)
```

Arguments

`filenames`	List of file names
`outfile`	Out file path
`template`	Template file path

Value

No return value, called for side effects

Examples

```
library(dplyr, warn.conflicts = FALSE)

data <- list(
  adsl = eg_adsl,
  adae = eg_adae,
  adtte = eg_adtte
)

filters::load_filters(
  yaml_file = system.file("filters.yml", package = "autoslider.core"),
  overwrite = TRUE
)

## For this example the outputs will be saved in a temporary directory. In a
## production run this should be the reporting event's 'output' folder instead.
output_dir <- tempdir()

spec_file <- system.file("spec.yml", package = "autoslider.core")
read_spec(spec_file) %>%
  filter_spec(program == "t_dm_slide") %>%
  generate_outputs(datasets = data) %>%
  decorate_outputs() %>%
  save_outputs(outfolder = output_dir)

slides_from_rds(list.files(file.path(output_dir, "t_dm_slide_FAS.rds")))
```

---

slides_preview *Generate flextable for preview first page*

Description

Generate flextable for preview first page

Usage

```
slides_preview(x)
```

Arguments

x	rtables or data.frame

Value

A flextable or a ggplot object depending to the input.

Examples

```
# Example 1. preview table
library(dplyr)
adsl <- eg_adsl
t_dm_slide(adsl, "TRT01P", c("SEX", "AGE")) %>% slides_preview()
```

`s_surv_time_1` *survival time afun*

Description

survival time afun

Usage

```
s_surv_time_1(df, .var, is_event, control = control_surv_time())
```

Arguments

`df`	data
`.var`	variable of interest
`is_event`	vector indicating event
`control`	‘control_surv_time()‘ by default

Value

A function suitable for use in rtables::analyze() with element selection, reformatting, and relabeling performed automatically.

`table_to_slide` *Add decorated flextable to slides*

Description

Add decorated flextable to slides

Usage

```
table_to_slide(
  ppt,
  content,
  decor = TRUE,
  table_loc = ph_location_type("body"),
  ...
)
```

Arguments

ppt	Slide
content	Content to be added
decor	Should table be decorated
table_loc	Table location
...	additional arguments

Value

Slide with added content

---

to_vector *Convert list of numbers to vectors*

Description

Convert list of numbers to vectors

Usage

```
to_vector(num_list)
```

Arguments

num_list	list of numbers

Value

No return value, called for side effects

---

trial *Cached trial*

Description

Cached trial data

Usage

```
trial
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 200 rows and 8 columns.

trim_perc *Format of xx.xx (xx.x)*

Description

Format of xx.xx (xx.x)

Usage

```
trim_perc(x, output)
```

Arguments

x	input array
output	output handle

Value

formatted values

trim_perc1 *Format of xx.xx (xx.xx)*

Description

Format of xx.xx (xx.xx)

Usage

```
trim_perc1(x, output)
```

Arguments

x	input array
output	output handle

Value

formatted values

t_aesi_slide *Table of AEs of Special Interest adapted from https://insightsengineering.github.io/tlg-catalog/stable/tables/adverse-events/aet01_aesi.html*

Description

Table of AEs of Special Interest adapted from https://insightsengineering.github.io/tlg-catalog/stable/tables/adverse-events/aet01_aesi.html

Usage

```
t_aesi_slide(adsl, adae, aesi, arm = "ACTARM", grad_var = "AETOXGR")
```

Arguments

adsl	ADSL data set, dataframe
adae	ADAE data set, dataframe.
aesi	AESI variable which will act as a filter to select the rows required to create the table. An example of AESI variable is CQ01NAM.
arm	Arm variable, character, ‘"ACTARM"‘ by default.
grad_var	Grading variable, character, ‘"AETOXGR"‘ by default.

Value

rtables object

Author(s)

Kai Xiang Lim (‘limk43‘)

Examples

```
library(dplyr)
adsl <- eg_adsl
adae <- eg_adae
adae_atoxgr <- adae %>% dplyr::mutate(ATOXGR = AETOXGR)
t_aesi_slide(adsl, adae, aesi = "CQ01NAM")
t_aesi_slide(adsl, adae, aesi = "CQ01NAM", arm = "ARM", grad_var = "AESEV")
t_aesi_slide(adsl, adae_atoxgr, aesi = "CQ01NAM", grad_var = "ATOXGR")
```

`t_ae_pt_diff_slide` *Adverse event table*

Description

Adverse event table

Usage

```
t_ae_pt_diff_slide(
  adsl,
  adae,
  arm = "TRT01A",
  cutoff = NA,
  split_by_study = FALSE,
  side_by_side = NULL
)
```

Arguments

adsl	ADSL data set, dataframe
adae	ADAE data set, dataframe
arm	Arm variable, character, "‘TRT01A" by default.
cutoff	Cutoff threshold
split_by_study	Split by study, building structured header for tables
side_by_side	"GlobalAsia" or "GlobalAsiaChina" to define the side by side requirement

Value

rtables object

Note

* Default arm variables are set to ‘"TRT01A"‘ for safety output, and ‘"TRT01P"‘ for efficacy output

Examples

```
library(dplyr)
adsl <- eg_adsl %>%
  dplyr::mutate(TRT01A = factor(TRT01A, levels = c("A: Drug X", "B: Placebo")))
adae <- eg_adae %>%
  dplyr::mutate(
    TRT01A = factor(TRT01A, levels = c("A: Drug X", "B: Placebo")),
    ATOXGR = AETOXGR
  )
out <- t_ae_pt_diff_slide(adsl, adae, "TRT01A", 2)
print(out)
generate_slides(out, paste0(tempdir(), "/ae_diff.pptx"))
```

t_ae_pt_slide *Adverse event table*

Description

Adverse event table

Usage

```
t_ae_pt_slide(
  adsl,
  adae,
  arm = "TRT01A",
  cutoff = NA,
  prune_by_total = FALSE,
  split_by_study = FALSE,
  side_by_side = NULL
)
```

Arguments

adsl	ADSL data set, dataframe
adae	ADAE data set, dataframe
arm	Arm variable, character, "‘TRT01A" by default.
cutoff	Cutoff threshold
prune_by_total	Prune according total column
split_by_study	Split by study, building structured header for tables
side_by_side	A logical value indicating whether to display the data side by side.

Value

rtables object

Note

* Default arm variables are set to ‘"TRT01A"‘ for safety output, and ‘"TRT01P"‘ for efficacy output

Examples

```
library(dplyr)
# Example 1
adsl <- eg_adsl %>%
  dplyr::mutate(TRT01A = factor(TRT01A, levels = c("A: Drug X", "B: Placebo")))
adae <- eg_adae %>%
  dplyr::mutate(
    TRT01A = factor(TRT01A, levels = c("A: Drug X", "B: Placebo")),
```

```
    ATOXGR = AETOXGR
  )
out <- t_ae_pt_slide(adsl, adae, "TRT01A", 2)
print(out)
generate_slides(out, paste0(tempdir(), "/ae.pptx"))

# Example 2, prune by total column
out2 <- t_ae_pt_slide(adsl, adae, "TRT01A", 25, prune_by_total = TRUE)
print(out2)
generate_slides(out, paste0(tempdir(), "/ae2.pptx"))
```

`t_ae_pt_soc_diff_slide` *Adverse event table*

Description

Adverse event table

Usage

```
t_ae_pt_soc_diff_slide(
  adsl,
  adae,
  arm = "TRT01A",
  cutoff = NA,
  split_by_study = FALSE,
  side_by_side = NULL
)
```

Arguments

`adsl`	ADSL data set, dataframe
`adae`	ADAE data set, dataframe
`arm`	Arm variable, character, "‘TRT01A" by default.
`cutoff`	Cutoff threshold
`split_by_study`	Split by study, building structured header for tables
`side_by_side`	"GlobalAsia" or "GlobalAsiaChina" to define the side by side requirement

Value

rtables object

Note

* Default arm variables are set to ‘"TRT01A"‘ for safety output, and ‘"TRT01P"‘ for efficacy output

Examples

```
library(dplyr)
adsl <- eg_adsl %>%
  dplyr::mutate(TRT01A = factor(TRT01A, levels = c("A: Drug X", "B: Placebo")))
adae <- eg_adae %>%
  dplyr::mutate(
    TRT01A = factor(TRT01A, levels = c("A: Drug X", "B: Placebo")),
    ATOXGR = AETOXGR
  )
out <- t_ae_pt_soc_diff_slide(adsl, adae, "TRT01A", 2)
print(out)
generate_slides(out, paste0(tempdir(), "/ae_diff.pptx"))
```

---

t_ae_pt_soc_slide *Adverse event table*

Description

Adverse event table

Usage

```
t_ae_pt_soc_slide(
  adsl,
  adae,
  arm,
  cutoff = NA,
  prune_by_total = FALSE,
  split_by_study = FALSE,
  side_by_side = NULL
)
```

Arguments

adsl	ADSL data set, dataframe
adae	ADAE data set, dataframe
arm	Arm variable, character
cutoff	Cutoff threshold
prune_by_total	Prune according total column
split_by_study	Split by study, building structured header for tables
side_by_side	"GlobalAsia" or "GlobalAsiaChina" to define the side by side requirement

Value

rtables object

Examples

```
library(dplyr)
# Example 1
adsl <- eg_adsl %>%
  dplyr::mutate(TRT01A = factor(TRT01A, levels = c("A: Drug X", "B: Placebo")))
adae <- eg_adae %>%
  dplyr::mutate(
    TRT01A = factor(TRT01A, levels = c("A: Drug X", "B: Placebo")),
    ATOXGR = AETOXGR
  )
out <- t_ae_pt_soc_slide(adsl, adae, "TRT01A", 2)
print(out)
generate_slides(out, paste0(tempdir(), "/ae.pptx"))


# Example 2, prune by total column
out2 <- t_ae_pt_soc_slide(adsl, adae, "TRT01A", 25, prune_by_total = TRUE)
print(out2)
generate_slides(out2, paste0(tempdir(), "/ae2.pptx"))
```

---

t_ae_slide *Adverse event table*

Description

Adverse event table

Usage

```
t_ae_slide(
  adsl,
  adae,
  arm = "TRT01A",
  split_by_study = FALSE,
  side_by_side = NULL
)
```

Arguments

adsl	ADSL data set, dataframe
adae	ADAE data set, dataframe
arm	Arm variable, character, "'TRT01A" by default.
split_by_study	Split by study, building structured header for tables
side_by_side	should table be displayed side by side

Value

rtables object

Note

* Default arm variables are set to ‘"TRT01A"‘ for safety output, and ‘"TRT01P"‘ for efficacy output

Examples

```
library(dplyr)
adsl <- eg_adsl %>%
  dplyr::mutate(TRT01A = factor(TRT01A, levels = c("A: Drug X", "B: Placebo")))
adae <- eg_adae %>%
  dplyr::mutate(
    TRT01A = factor(TRT01A, levels = c("A: Drug X", "B: Placebo")),
    ATOXGR = AETOXGR
  )
out <- t_ae_slide(adsl, adae, "TRT01A")
print(out)
generate_slides(out, paste0(tempdir(), "/ae.pptx"))
```

---

t_ae_summ_slide *Adverse event summary table*

Description

Adverse event summary table

Usage

```
t_ae_summ_slide(
  adsl,
  adae,
  arm = "TRT01A",
  dose_adjust_flags = NA,
  dose_adjust_labels = NA,
  gr34_highest_grade_only = TRUE
)
```

Arguments

adsl	ADSL dataset, dataframe
adae	ADAE dataset, dataframe
arm	Arm variable, character, "‘TRT01A" by default.
dose_adjust_flags	Character or a vector of characters. Each character is a variable name in adae dataset. These variables are Logical vectors which flag AEs leading to dose adjustment, such as drug discontinuation, dose interruption and reduction. The flag can be related to any drug, or a specific drug.

dose_adjust_labels
: Character or a vector of characters. Each character represents a label displayed in the AE summary table (e.g. AE leading to discontinuation from drug X). The order of the labels should match the order of variable names in dose_adjust_flags.

gr34_highest_grade_only
: A logical value. Default is TRUE, such that only patients with the highest AE grade as 3 or 4 are included for the count of the "Grade 3-4 AE" and "Treatment-related Grade 3-4 AE" ; set it to FALSE if you want to include patients with the highest AE grade as 5.

Value

an rtables object

Examples

```
library(dplyr)
ADSL <- eg_adsl
ADAE <- eg_adae

ADAE <- ADAE %>%
  dplyr::mutate(ATOXGR = AETOXGR)
t_ae_summ_slide(adsl = ADSL, adae = ADAE)

# add flag for ae leading to dose reduction
ADAE$reduce_flg <- ifelse(ADAE$AEACN == "DOSE REDUCED", TRUE, FALSE)
t_ae_summ_slide(
  adsl = ADSL, adae = ADAE,
  dose_adjust_flags = c("reduce_flg"),
  dose_adjust_labels = c("AE leading to dose reduction of drug X")
)
# add flgs for ae leading to dose reduction, drug withdraw and drug interruption
ADAE$withdraw_flg <- ifelse(ADAE$AEACN == "DRUG WITHDRAWN", TRUE, FALSE)
ADAE$interrup_flg <- ifelse(ADAE$AEACN == "DRUG INTERRUPTED", TRUE, FALSE)
out <- t_ae_summ_slide(
  adsl = ADSL, adae = ADAE, arm = "TRT01A",
  dose_adjust_flags = c("withdraw_flg", "reduce_flg", "interrup_flg"),
  dose_adjust_labels = c(
    "AE leading to discontinuation from drug X",
    "AE leading to drug X reduction",
    "AE leading to drug X interruption"
  )
)
print(out)
generate_slides(out, paste0(tempdir(), "/ae_summary.pptx"))
```

---

t_dd_slide *Death table*

Description

Death table

Usage

```
t_dd_slide(adsl, arm = "TRT01A", split_by_study = FALSE, side_by_side = NULL)
```

Arguments

adsl	ADSL data set, dataframe
arm	Arm variable, character, "‘TRT01A" by default.
split_by_study	Split by study, building structured header for tables
side_by_side	used for studies in China. "GlobalAsia" or "GlobalAsiaChina" to define the side by side requirement.

Value

rtables object

Note

* Default arm variables are set to ‘"TRT01A"‘ for safety output, and ‘"TRT01P"‘ for efficacy output

Examples

```
library(dplyr)
adsl <- eg_adsl %>% preprocess_t_dd()
out1 <- t_dd_slide(adsl, "TRT01A")
print(out1)
generate_slides(out1, paste0(tempdir(), "/dd.pptx"))

out2 <- t_dd_slide(adsl, "TRT01A", split_by_study = TRUE)
print(out2)
```

---

t_dm_slide *Demographic table*

Description

Demographic table

Usage

```
t_dm_slide(
  adsl,
  arm = "TRT01P",
  vars = c("AGE", "SEX", "RACE"),
  stats = c("median", "range", "count_fraction"),
  split_by_study = FALSE,
  side_by_side = NULL
)
```

Arguments

adsl	ADSL data set, dataframe
arm	Arm variable, character, "'TRT01P" by default.
vars	Characters of variables
stats	see '.stats' from [tern::analyze_vars()]
split_by_study	Split by study, building structured header for tables
side_by_side	"GlobalAsia" or "GlobalAsiaChina" to define the side by side requirement

Value

rtables object

Note

* Default arm variables are set to '"TRT01A"' for safety output, and '"TRT01P"' for efficacy output

Examples

```
library(dplyr)
adsl <- eg_adsl
out1 <- t_dm_slide(adsl, "TRT01P", c("SEX", "AGE", "RACE", "ETHNIC", "COUNTRY"))
print(out1)
generate_slides(out1, paste0(tempdir(), "/dm.pptx"))

out2 <- t_dm_slide(adsl, "TRT01P", c("SEX", "AGE", "RACE", "ETHNIC", "COUNTRY"),
  split_by_study = TRUE
)
print(out2)
```

t_dor_slide *DOR table*

Description

DOR table

Usage

```
t_dor_slide(adsl, adtte, arm = "TRT01P", refgroup = NULL)
```

Arguments

adsl	ADSL dataset
adtte	ADTTE dataset
arm	Arm variable, character, "'TRT01P" by default.
refgroup	Reference group

Value

An 'rtables' object

Note

* Default arm variables are set to '"TRT01A"' for safety output, and '"TRT01P"' for efficacy output

Examples

```
library(dplyr)
adsl <- eg_adsl %>%
 dplyr::mutate(TRT01P = factor(TRT01P, levels = c("A: Drug X", "B: Placebo", "C: Combination")))
adtte <- eg_adtte %>%
  dplyr::filter(PARAMCD == "OS") %>%
 dplyr::mutate(TRT01P = factor(TRT01P, levels = c("A: Drug X", "B: Placebo", "C: Combination")))
out <- t_dor_slide(adsl, adtte)
print(out)
generate_slides(out, paste0(tempdir(), "/dor.pptx"))
```

t_ds_slide *Discontinue table*

Description

Discontinue table

Usage

```
t_ds_slide(adsl, arm = "TRT01P", split_by_study = FALSE, side_by_side = NULL)
```

Arguments

adsl	ADSL data
arm	Arm variable, character, "'TRT01P" by default.
split_by_study	Split by study, building structured header for tables
side_by_side	"GlobalAsia" or "GlobalAsiaChina" to define the side by side requirement

Note

* Default arm variables are set to '"TRT01A"' for safety output, and '"TRT01P"' for efficacy output

Examples

```
library(dplyr)
adsl <- eg_adsl %>%
 mutate(DISTRTFL = sample(c("Y", "N"), size = nrow(eg_adsl), replace = TRUE, prob = c(.1, .9))) %>%
  preprocess_t_ds()
out1 <- t_ds_slide(adsl, "TRT01P")
print(out1)
generate_slides(out1, paste0(tempdir(), "/ds.pptx"))

out2 <- t_ds_slide(adsl, "TRT01P", split_by_study = TRUE)
print(out2)
```

use_template *[EXPERIMENTAL] Create new output function based on a template.*

Description

We have separate templates for listings, tables, and graphs. There is also a template to set up the 'run_all' script.

Usage

```
use_template(
  template = "t_dm_slide",
  function_name = "default",
  save_path = "./programs/R",
  overwrite = FALSE,
  open = interactive(),
  package = "autoslider.core"
)
```

Arguments

template	Must be one of ‘list_all_templates(package = "autoslider.core")‘.
function_name	Name of the output function you want to create. Defaults to "default".
save_path	Path to save the function. Defaults to "./programs/R".
overwrite	Whether to overwrite an existing file.
open	Whether to open the file after creation.
package	Which package to search for the template file. Defaults to "autoslider.core".

Details

Use ‘list_all_templates(package = "autoslider.core")‘ to discover which templates are available.

Value

No return value. Called for side effects (writes a file).

Examples

```
if (interactive()) {
  use_template("t_dm_slide", function_name = "my_table", package = "autoslider.core")
}
```

Index