Package ‘rlistings’

May 25, 2023

**Title**  Clinical Trial Style Data Readout Listings

**Version**  0.2.2

**Date**  2023-05-02

**Description**  Listings are often part of the submission of clinical trial data in regulatory settings. We provide a framework for the specific formatting features often used when displaying large datasets in that context.

**License**  Apache License 2.0

**URL**  [https://github.com/insightsengineering/rlistings](https://github.com/insightsengineering/rlistings)

**BugReports**  [https://github.com/insightsengineering/rlistings/issues](https://github.com/insightsengineering/rlistings/issues)

**Depends**  formatters (&gt;= 0.5.0), methods, tibble

**Imports**  checkmate, grDevices, grid, stats, utils

**Suggests**  dplyr, knitr (&gt;= 1.34), lifecycle, testthat

**VignetteBuilder**  knitr

**Config/Needs/website**  insightsengineering/nesttemplate

**Encoding**  UTF-8

**Language**  en-US

**RoxygenNote**  7.2.3

**NeedsCompilation**  no

**Author**  Gabriel Becker [aut, cre],
Adrian Waddell [aut],
Joe Zhu [aut],
Davide Garolini [ctb],
Emily de la Rua [ctb],
Abinaya Yogasekaram [ctb],
F. Hoffmann-La Roche AG [cph, fnd]

**Maintainer**  Gabriel Becker &lt;gabembecker@gmail.com&gt;

**Repository**  CRAN

**Date/Publication**  2023-05-25 16:40:02 UTC
as_listing

Create a Listing from a data.frame or tibble

Description

[Experimental]

Creates listings by using cols and key.cols to produce a compact and elegant representation of the data.frame or tibble in input.

Usage

as_listing(
  df,
  key_cols = names(df)[1],
  disp_cols = NULL,
  non_disp_cols = NULL,
  main_title = NULL,
  subtitles = NULL,
  main_footer = NULL,
  prov_footer = NULL
)

as_keycol(vec)

is_keycol(vec)

get_keycols(df)

listing_dispcols(df)

add_listing_dispcol(df, new)

listing_dispcols(df) <- value

add_listing_col(df, name, fun = NULL, format = NULL, na_str = "-")
as_listing

Arguments

df data.frame or listing_df. The (non-listing) data.frame to be converted to a listing or the listing_df to be modified.

key_cols character. Names of columns which should be treated as key columns when rendering the listing. Key columns allow you to group repeat occurrences.

disp_cols character or NULL. Names of non-key columns which should be displayed when the listing is rendered. Defaults to all columns of df not named in key_cols or non_disp_cols.

non_disp_cols character or NULL. Names of non-key columns to be excluded as display columns. All other non-key columns are then treated as display columns. Invalid if disp_cols is non-NULL.

main_title character(1) or NULL. The main title for the listing, or NULL (the default). Must be length 1 non-NULL.

subtitles character or NULL. A vector of subtitle(s) for the listing, or NULL (the default).

main_footer character or NULL. A vector of main footer lines for the listing, or NULL (the default).

prov_footer character or NULL. A vector of provenance strings for the listing, or NULL (the default). Each string element is placed on a new line.

vec any. A column vector from a listing_df to be annotated as a key column.

new character. Names of columns to be added to the set of display columns.

value character. New value.

name character(1). Name of the existing or new column to be displayed when the listing is rendered.

fun function or NULL. A function which accepts df and returns the vector for a new column, which is added to df as name, or NULL if marking an existing column as a listing column.

format character(1) or function. The format label (string) or formatter function to apply to x.

na_str character(1). String that should be displayed when the value of x is missing. Defaults to "NA".

Details

At its core, a listing_df object is a tbl_df object with a customized print method and support for the formatting and pagination machinery provided by the formatters package.

listing_df objects have two ‘special’ types of columns: key columns and display columns.

Key columns act as indexes, which means a number of things in practice.

All key columns are also display columns.

listing_df objects are always sorted by their set of key_columns at creation time. Any listing_df object which is not sorted by its full set of key columns (e.g., one whose rows have been reordered explicitly creation) is invalid and the behavior when rendering or paginating that object is undefined.

Each value of a key column is printed only once per page and per unique combination of values for all higher-priority (i.e., to the left of it) key columns. Locations where a repeated value would have
been printed within a key column for the same higher-priority-key combination on the same page
are rendered as empty space. Note, determination of which elements to display within a key column
at rendering is based on the underlying value; any non-default formatting applied to the column has
no effect on this behavior.

Display columns are columns which should be rendered, but are not key columns. By default this
is all non-key columns in the incoming data, but in need not be. Columns in the underlying data
which are neither key nor display columns remain within the object available for computations but
are not rendered during printing or export of the listing.

Value

A listing_df object, sorted by the key columns.

df, with name created (if necessary) and marked for display during rendering.

Examples

dat <- ex_adae

# This example demonstrates the listing with key_cols (values are grouped by USUBJID) and
# multiple lines in prov_footer
 lsting <- as_listing(dat[1:25, ],
   key_cols = c("USUBJID", "AESOC"),
   main_title = "Example Title for Listing",
   subtitles = "This is the subtitle for this Adverse Events Table",
   main_footer = "Main footer for the listing",
   prov_footer = c(
     "You can even add a subfooter", "Second element is place on a new line",
     "Third string"
   )
 )
%%
 add_listing_col("AETOXGR") %>%
 add_listing_col("BMRKR1", format = "xx.x") %>%
 add_listing_col("AESER / AREL", fun = function(df) paste(df$AESER, df$AREL, sep = " / "))

mat <- matrix_form(lsting)

cat(toString(mat))

cat(toString(mat))

# This example demonstrates the listing table without key_cols
# and specifying the cols with disp_cols.
 dat <- ex_adae
 lsting <- as_listing(dat[1:25, ], disp_cols = c("USUBJID", "AESOC", "RACE", "AETOXGR", "BMRKR1"))

mat <- matrix_form(lsting)

cat(toString(mat))

cat(toString(mat))
**make_row_df, listing_df-method**

*Make pagination dataframe for a listing*

**Description**

Make pagination dataframe for a listing

**Usage**

```r
## S4 method for signature 'listing_df'
make_row_df(
    tt, 
    colwidths = NULL, 
    visible_only = TRUE, 
    rownum = 0, 
    indent = 0L, 
    path = character(), 
    incontent = FALSE, 
    repr_ext = 0L, 
    repr_inds = integer(), 
    sibpos = NA_integer_, 
    nsibs = NA_integer_,
)
```

**Arguments**

- **tt**: listing_df. The listing to be rendered
- **colwidths**: numeric. Internal detail do not set manually.
- **visible_only**: logical(1). Ignored, as listings do not have non-visible structural elements.
- **rownum**: numeric(1). Internal detail do not set manually.
- **indent**: integer(1). Internal detail do not set manually.
- **path**: character. Path to the (sub)table represented by tt. Defaults to character()
- **incontent**: logical(1). Internal detail do not set manually.
- **repr_ext**: integer(1). Internal detail do not set manually.
- **repr_inds**: integer. Internal detail do not set manually.
- **sibpos**: integer(1). Internal detail do not set manually.
- **nsibs**: integer(1). Internal detail do not set manually.

**Value**

a data.frame with pagination information.
See Also

make_row_df

Examples

lsting <- as_listing(mtcars)
mf <- matrix_form(lsting)

matrix_form,listing_df-method

Transform rtable to a list of matrices which can be used for outputting

Description

Although rtables are represented as a tree data structure when outputting the table to ASCII or HTML it is useful to map the rtable to an in between state with the formatted cells in a matrix form.

Usage

## S4 method for signature 'listing_df'
matrix_form(obj, indent_rownames = FALSE)

Arguments

obj ANY. Object to be transformed into a ready-to-render form (a MatrixPrintForm object)
indent_rownames logical(1). Silently ignored, as listings do not have row names nor indenting structure.

Details

The strings in the return object are defined as follows: row labels are those determined by summarize_rows and cell values are determined using get_formatted_cells. (Column labels are calculated using a non-exported internal function.

Value

a MatrixPrintForm object

See Also

formatters::matrix_form()
paginate_listing

Examples

```r
lsting <- as_listing(mtcars)
mf <- matrix_form(lsting)
```

**paginate_listing**  
**Paginate listings**

**Description**

[Experimental]

Pagination of a listing. This can be vertical for long listings with many rows or horizontal if there are many columns.

**Usage**

```r
paginate_listing(
  lsting,
  page_type = "letter",
  font_family = "Courier",
  font_size = 8,
  lineheight = 1,
  landscape = FALSE,
  pg_width = NULL,
  pg_height = NULL,
  margins = c(top = 0.5, bottom = 0.5, left = 0.75, right = 0.75),
  lpp = NA_integer_,
  cpp = NA_integer_,
  colwidths = propose_column_widths(lsting),
  tf_wrap = !is.null(max_width),
  max_width = NULL,
  verbose = FALSE
)
```

**Arguments**

- **lsting**: listing_df. The listing to paginate.
- **page_type**: character(1). Name of a page type. See page_types. Ignored when pg_width and pg_height are set directly.
- **font_family**: character(1). Name of a font family. An error will be thrown if the family named is not monospaced. Defaults to Courier.
- **font_size**: numeric(1). Font size, defaults to 12.
- **lineheight**: numeric(1). Line height, defaults to 1.
paginate_listing

landscape logical(1). Should the dimensions of page_type be inverted for landscape? Defaults to FALSE, ignored when pg_width and pg_height are set directly.

pg_width numeric(1). Page width in inches.

pg_height numeric(1). Page height in inches.

margins numeric(4). Named numeric vector containing 'bottom', 'left', 'top', and 'right' margins in inches. Defaults to .5 inches for both vertical margins and .75 for both horizontal margins.

lpp numeric(1) or NULL. Number of row lines (not counting titles and footers) to have per page. Standard is 70 while NULL disables vertical pagination.

cpp numeric(1) or NULL. Width (in characters) of the pages for horizontal pagination. NULL (the default) indicates no horizontal pagination should be done.

colwidths numeric vector. Column widths (in characters) for use with vertical pagination.

tf_wrap logical(1). Should the texts for title, subtitle, and footnotes be wrapped?

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

verbose logical(1). Should additional informative messages about the search for pagination breaks be shown. Defaults to FALSE.

Value

A list of listings’ objects that are meant to be on separated pages. For pag_tt_indices a list of paginated-groups of row-indices of listing.

for paginate_listing a list containing separate listing_df objects for each page, for pag_listing_indices, a list of indices in the direction being paginated corresponding to the individual pages in that dimension.

Examples

dat <- ex_adae
lsting <- as_listing(dat[1:25, ], disp_cols = c("USUBJID", "AESOC", "RACE", "AETOXGR", "BMRKR1"))

mat <- matrix_form(lsting)
cat(toString(mat))
paginate_listing(lsting, lpp = 10)
paginate_listing(lsting, cpp = 100, lpp = 40)
paginate_listing(lsting, cpp = 80, lpp = 40, verbose = TRUE)
pag_listing_indices

Description

These functions are defunct and their symbols will be removed entirely in a future release.

Usage

```r
pag_listing_indices(
  lsting,
  lpp = 15,
  colwidths = NULL,
  max_width = NULL,
  verbose = FALSE
)
```

Arguments

- `lsting`: listing_df. The listing to paginate.
- `lpp`: numeric(1) or NULL. Number of row lines (not counting titles and footers) to have per page. Standard is 70 while NULL disables vertical pagination.
- `colwidths`: numeric vector. Column widths (in characters) for use with vertical pagination.
- `max_width`: integer(1), character(1) or NULL. Width that title and footer (including footnotes) materials should be word-wrapped to. If NULL, it is set to the current print width of the session (`getOption("width")`). If set to "auto", the width of the table (plus any table inset) is used. Ignored completely if `tf_wrap` is FALSE.
- `verbose`: logical(1). Should additional informative messages about the search for pagination breaks be shown. Defaults to FALSE.
Index

add_listing_col (as_listing), 2
add_listing_dispcol (as_listing), 2
as_keycol (as_listing), 2
as_listing, 2
formatters::matrix_form(), 6
get_keycols (as_listing), 2
is_keycol (as_listing), 2
listing_dispcols (as_listing), 2
listing_dispcols<- (as_listing), 2
make_row_df, 6
make_row_df, listing_df-method, 5
matrix_form, listing_df-method, 6
pag_listing_indices, 9
paginate_listing, 7