Package ‘ftExtra’

September 29, 2023

Title Extensions for ‘Flextable’
Version 0.6.1
Date 2023-09-28
Maintainer Atsushi Yasumoto <atusy.rpkg@gmail.com>
Description Build display tables easily by extending the functionality of the ‘flextable’ package. Features include spanning header, grouping rows, parsing markdown and so on.
License MIT + file LICENSE
BugReports https://github.com/atusy/ftExtra/issues
Imports dplyr (>= 1.0.0), jsonlite, flextable (>= 0.8.3), tidyr, purrr
       (>= 1.0.0), magrittr, rmarkdown, rlang, stringr, tibble,
       tidyselect (>= 1.1.0), xfun, yaml, vctrs
Suggests testthat (>= 2.1.0), knitr, qpdf
VignetteBuilder knitr
Encoding UTF-8
Language en-US
RoxygenNote 7.2.3
SystemRequirements pandoc (>= 2.0.6) - http://pandoc.org
Config/testthat/edition 3
NeedsCompilation no
Author Atsushi Yasumoto [aut, cph, cre]
       (<https://orcid.org/0000-0002-8335-495X>),
       David Gohel [ctb],
       Romain François [ctb] (<https://orcid.org/0000-0002-2444-4226>),
       Tatsuya Shima [ctb]
Repository CRAN
Date/Publication 2023-09-29 03:40:02 UTC
as_flextable_methods

Method to transform objects into flextables

Description
This is a convenient function to let users create flextable bindings from any objects. Users should consult documentation of corresponding method to understand the details and see what arguments can be used.

Usage

```r
## S3 method for class 'grouped_df'
as_flextable(
  x,
  groups_to = c("titles", "merged", "asis"),
  groups_pos = c("left", "asis"),
  groups_arrange = NULL,
  ...
)
```

```r
## S3 method for class 'data.frame'
as_flextable(x, col_keys = names(x), ...)
```

Arguments

- `x` object to be transformed as flextable
- `groups_to` One of `titles`, `merged`, or `asis`. See examples and vignette("group-rows") for the result.
- `groups_pos` When `groups_to = "merged"`, grouping columns are reordered according to `group_pos`. Choices are `left` (default) or `asis`.
- `groups_arrange` TRUE automatically arranges grouping columns by `dplyr::arrange()`. Specify `FALSE` to keep the arrangement of the input data frame. The default value is `NULL` which implies `FALSE` to keep the backward compatibility, but will be `TRUE` in the future.
- `...` arguments for custom methods
col_keys columns names/keys to display. If some column names are not in the dataset, they will be added as blank columns by default.

See Also

Other as_flextable methods: as_flextable.data.frame(), as_flextable.gam(), as_flextable.glm(), as_flextable.grouped_data(), as_flextable.htest(), as_flextable.kmeans(), as_flextable.lm(), as_flextable.merMod(), as_flextable.pam(), as_flextable.summarizer(), as_flextable.table(), as_flextable.tabular(), as_flextable.tabulator(), as_flextable.xtable()

Examples

# For grouped_df
grouped_df <- iris %>%
  dplyr::group_by(Species) %>%
  dplyr::slice(1, 2)

as_flextable(grouped_df, groups_to = "titles")
as_flextable(grouped_df, groups_to = "titles", hide_grouplabel = TRUE)
as_flextable(grouped_df, groups_to = "merged")
as_flextable(grouped_df, groups_to = "asis")
# For data.frame
iris %>%
  head() %>%
  as_flextable()

---

`as_paragraph_md` Convert a character vector into markdown paragraph(s)

**Description**

Parse markdown cells and returns the "paragraph" object.

**Usage**

```r
as_paragraph_md(
  x,
  auto_color_link = "blue",
  md_extensions = NULL,
  pandoc_args = NULL,
  metadata = rmarkdown::metadata,
  replace_na = "",
  .from = "markdown+autolink_bare_uris-raw_html-raw_attribute",
  .footnote_options = NULL,
  ...
)
```
Arguments

- **x**: A character vector.
- **auto_color_link**: A color of the link texts.
- **md_extensions**: Pandoc’s extensions. Although it is prefixed with "md", extensions for any formats specified to .from can be used. See [https://www.pandoc.org/MANUAL.html#extensions](https://www.pandoc.org/MANUAL.html#extensions) for details.
- **pandoc_args**: Additional command line options to pass to pandoc.
- **metadata**: A list of metadata, typically the parsed result of the YAML front matter (default: rmarkdown::metadata). This value is used iff the .from argument specifies the input format that supports the YAML metadata blocks.
- **replace_na**: A value to replace NA (default = "").
- **.from**: Pandoc’s --from argument (default: 'markdown+autolink_bare_uris').
- **.footnote_options**: Options for footnotes generated by footnote_options().
- **...**: Arguments passed to internal functions.

Examples

```r
if (rmarkdown::pandoc_available("2.0.6")) {
  library(flextable)
  ft <- flextable(
    data.frame(
      x = c("**foo** bar", "***baz***", "*qux*"),
      stringsAsFactors = FALSE
    )
  )
  ft <- compose(ft, j = "x", i = 1:2, value = as_paragraph_md(x))
  autofit(ft)
}
```

---

**colformat_md**

Format character columns as markdown text

**Description**

Format character columns as markdown text

**Usage**

```r
colformat_md(
  x,
  j = where(is.character),
  part = c("body", "header", "all"),
  auto_color_link = "blue",
```
Arguments

x  A flextable object

j  Columns to be treated as markdown texts. Selection can be done by the semantics of dplyr::select().

colnames

part  One of "body", "header", and "all". If "all", formatting proceeds in the order of "header" and "body".

auto_color_link  A color of the link texts.

md_extensions  Pandoc’s extensions. Although it is prefixed with "md", extensions for any formats specified to .from can be used. See https://www.pandoc.org/Manual.html#extensions for details.

pandoc_args  Additional command line options to pass to pandoc

metadata  A list of metadata, typically the parsed result of the YAML front matter (default: rmarkdown::metadata). This value is used iff the .from argument specifies the input format that supports the YAML metadata blocks.

replace_na  A value to replace NA (default = "").

.from  Pandoc’s --from argument (default: 'markdown+autolink_bare_uri').

.footnote_options  Options for footnotes generated by footnote_options().

.sep  A separator of paragraphs (default: "\n\n")

Examples

if (rmarkdown::pandoc_available("2.0.6")) {
  d <- data.frame(
    x = c("**bold**", "*italic*"),
    y = c("^superscript^", "~subscript~"),
    z = c("***ft*~Extra* is*", "*Cool*")
  )
  colformat_md(flextable::flextable(d))
}
## Options for footnotes

### Description

Configure options for footnotes.

### Usage

```r
footnote_options(
  ref = c("1", "a", "A", "i", "I", "*"),
  prefix = "",
  suffix = "",
  start = 1L,
  max = 26L,
  inline = FALSE,
  sep = "; ",
)
```

### Arguments

- **ref**: A string or a function that defines symbols of footnote references. If the value is string, it must be one of the "1", "a", "A", "i", "I", or "*". If a function, keep in mind this is an experimental feature. It receives 3 parameters (n, part, and footer) and returns character vectors which will further be processed as markdown. See examples for the details.
- **prefix**, **suffix**: Pre- and suf-fixes for ref (default: ""). These parameters are used if and only if ref is a character.
- **start**: A starting number of footnotes.
- **max**: A max number of footnotes used only when ref is "a" or "A".
- **inline**: whether to add footnote on same line as previous footnote or not
- **sep**: used only when inline = TRUE, character string to use as a separator between footnotes.

### Value

An environment

### Examples

```r
# A example flextable with unprocessed markdown footnotes
ft <- flextable(tibble::tibble(
  "header1^[note a]" = c("x^[note 1]", "y"),
  "header2" = c("a", "b^[note 2]")
))
```
# Render all footnotes in the same format.
if (rmarkdown::pandoc_available("2.0.6")) {
    ft %>%
        colformat_md(
            part = "all",
            .footnote_options = footnote_options("1", start = 1L)
        )
}

# Use a user-defined function to format footnote symbols
if (rmarkdown::pandoc_available("2.0.6")) {
    # a function to format symbols of footnote references
    ref <- function(n, part, footer) {
        # Change symbols by context
        # - header: letters (a, b, c, ...)
        # - body: integers (1, 2, 3, ...)
        s <- if (part == "header") {
            letters[n]
        } else {
            as.character(n)
        }
    }
    # Suffix symbols with ": " (a colon and a space) in the footer
    if (footer) {
        return(paste0(s, ":\ "))
    }
    # Use superscript in the header and the body
    return(paste0("^^", s, "^^"))
}

# apply custom format of symbols
ft %>%
    # process header first
    colformat_md(
        part = "header", .footnote_options = footnote_options(ref = ref)
    ) %>%
    # process body next
    colformat_md(
        part = "body", .footnote_options = footnote_options(ref = ref)
    ) %>%
    # tweak width for visibility
    flextable::autofit(add_w = 0.2)
}

## span_header

**Span the header based on delimiters**

### Description

Span the header based on delimiters
Usage

split_header(x, sep = "\[_.\]", theme_fun = NULL, ...)

Arguments

- `x` A flextable object
- `sep` Separator between columns.
  - If character, `sep` is interpreted as a regular expression. The default value is a regular expression that matches any sequence of non-alphanumeric values.
  - If numeric, `sep` is interpreted as character positions to split at. Positive values start at 1 at the far-left of the string; negative values start at -1 at the far-right of the string. The length of `sep` should be one less than `into`.
- `theme_fun` A flextable theme function. When NULL (default), the value is resolved by `flextable::get_flextable_defaults()`.
- `...` Passed to `theme_fun`

Note

`split_header` is a rename of `separate_header` and the latter will be removed in the future release.

Examples

```r
iris %>%
  flextable() %>%
  span_header()
```

---

**split_header**

Split the header based on delimiters

Description

Split the header based on delimiters

Usage

split_header(x, sep = "\[_.\]", theme_fun = NULL, ...)

separate_header(x, sep = "\[_.\]", theme_fun = NULL, ...)

Arguments

- `x` A flextable object
with_blanks

sep
Separator between columns.
If character, sep is interpreted as a regular expression. The default value is a
regular expression that matches any sequence of non-alphanumeric values.
If numeric, sep is interpreted as character positions to split at. Positive values
start at 1 at the far-left of the string; negative value start at -1 at the far-right of
the string. The length of sep should be one less than into.

theme_fun
A flextable theme function. When NULL (default), the value is resolved by
flextable::get_flextable_defaults().

... Passed to theme_fun

Note
split_header is a rename of separate_header and the latter will be removed in the future release.

Examples
iris %>%
  flextable() %>%
  separate_header()

with_blanks Specify blank columns easily via col_keys

Description
Specify blank columns easily via col_keys

Usage
with_blanks(after = NULL, before = NULL)

Arguments
after, before Blank columns are added after/before the selected columns. Selections can be
done by the semantics of dplyr::select.

Examples
iris %>%
  as_flextable(col_keys = with_blanks(dplyr::ends_with("Width")))
Index

as_flexttable.data.frame, 3
as_flexttable.data.frame
  (as_flexttable_methods), 2
as_flexttable.gam, 3
as_flexttable.glm, 3
as_flexttable.grouped_data, 3
as_flexttable.grouped_df
  (as_flexttable_methods), 2
as_flexttable.htest, 3
as_flexttable.kmeans, 3
as_flexttable.lm, 3
as_flexttable.merMod, 3
as_flexttable.pam, 3
as_flexttable.summarizor, 3
as_flexttable.table, 3
as_flexttable.tabular, 3
as_flexttable.tabulator, 3
as_flexttable.xtable, 3
as_flexttable_methods, 2
as_paragraph_md, 3

colformat_md, 4
dplyr::arrange(), 2

footnote_options, 6

separate_header (split_header), 8
span_header, 7
split_header, 8

with_blanks, 9