Package ‘tablexlsx’

October 16, 2024

Type Package

Title Export Data Frames to Excel Workbook

Version 1.1.0

Description Collection of functions that allow to export data frames to excel workbook.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

URL https://ddotta.github.io/tablexlsx/,
https://github.com/ddotta/tablexlsx

BugReports https://github.com/ddotta/tablexlsx/issues

RoxygenNote 7.2.3

Depends R (>= 2.10)

Suggests rmarkdown, knitr, testthat (>= 3.0.0)

Config/testthat/edition 3

Imports magrittr, methods, openxlsx, cli

VignetteBuilder knitr

NeedsCompilation no

Author Damien Dotta [aut, cre],
Julien Blasco [aut]

Maintainer Damien Dotta <damien.dotta@live.fr>

Repository CRAN

Date/Publication 2024-10-16 13:50:02 UTC

Contents

add_table *Function that adds a data frame to an (existing) .xlsx workbook sheet*

Description

Function that adds a data frame to an (existing) .xlsx workbook sheet

Usage

```
add_table(
  Table,
  WbTitle,
  SheetTitle,
  TableTitle,
  StartRow = 1,
  StartCol = 1,
 FormatList = setNames(rep(list(Theme[[”character”]]), length(colnames(Table))),
    colnames(Table)),
  Theme = xls_theme_default(),
  HeightTableTitle = 2,
  TableFootnote1 = "",
  TableFootnote2 = "",
  TableFootnote3 = "",
  MergeCol = character(0),
  ByGroup = character(0),
  GroupName = FALSE,
  asTable = FALSE
)
```

Arguments

`Table` : data frame to be exported to the workbook sheet

`WbTitle` : workbook

`SheetTitle` : string used for the sheet’s name

`TableTitle` : string used for the data frame’s title

`StartRow` : export start line number in the sheet (by default 1)

`StartCol` : export start column number in the sheet (by default 1)

`FormatList` : list that indicates the format of each column of the data frame

`Theme` : styling theme, a named list of ‘openxlsx‘ Styles

`HeightTableTitle`
: multiplier (if needed) for the height of the title line (by default 2)

`TableFootnote1` : string for TableFootnote1

`TableFootnote2` : string for TableFootnote2

`TableFootnote3` : string for TableFootnote3

`MergeCol` : character vector that indicates the columns for which to merge the modalities

`ByGroup` character vector indicating the name of the columns by which to group

`GroupName` boolean indicating whether the name of the grouping variable should be written

`asTable` logical indicating if data should be written as an Excel Table (FALSE by default)

Value

excel wb object

`style` *An R list that contains the styles of each element for formatting data frames in excel files*

Description

An R list that contains the styles of each element for formatting data frames in excel files

Usage

```
style
```

Format

A list of several items

title Style for title

footnote1 Style for footnote1

footnote2 Style for footnote2

footnote3 Style for footnote3

col_header Style for col_header

character Style for columns in character format

number Style for columns in number format

decimal Style for columns in character format with decimals

percent Style for columns in percent format `toxlsx` *Convert R data frames to excel files*

Description

This function allows you to write R data frames given in the ‘object‘ argument to excel files located in the ‘path‘ directory. The function takes several arguments but the only two required are ‘object‘ and ‘path‘.
See examples gallery : <https://ddotta.github.io/tablexlsx/articles/aa-examples.html>

Usage

```
toxlsx(
  object,
  path,
  tosheet = list(),
  title = list(),
  columnstyle = list(default = NULL),
  theme = xls_theme_default(),
  footnote1 = list(),
  footnote2 = list(),
  footnote3 = list(),
  mergecol = NULL,
  bygroup = list(),
  groupname = FALSE,
  filename = "Export",
  asTable = FALSE,
  automaticopen = FALSE
)
```

Arguments

`object`	data.frame to be converted to excel
`path`	path to save excel file (either a directory name or a file name with full path)
`tosheet`	list of sheet names for each element of object. If omitted, sheets are named by default "Sheet 1", "Sheet 2"...
`title`	list of title for each element of object If omitted, title takes the name of the dataframe in ‘object‘
`columnstyle`	list of style for columns of each element of object Only useful if you want to customise the style of each column ‘
`theme`	styling theme, a named list of ‘openxlsx‘ Styles
`footnote1`	list of footnote1 for each element of object If omitted, no footnote1
`footnote2`	list of footnote2 for each element of object If omitted, no footnote2
`footnote3`	list of footnote3 for each element of object If omitted, no footnote3

`mergecol` list of character vectors that indicate the columns for which we want to merge the modalities

`bygroup` list of character vectors indicating the names of the columns by which to group

`groupname` list of booleans indicating whether the names of the grouping variables should be written

`filename` name for the excel file ("Export" by default). Ignored if ‘path‘ is a file name.

`asTable` logical indicating if data should be written as an Excel Table (FALSE by default)

`automaticopen` logical indicating if excel file should open automatically (FALSE by default)

Value

an excel file

Examples

```
# Simply export a data frame to an xlsx file
# For more examples, see examples gallery :
# https://ddotta.github.io/tablexlsx/articles/aa-examples.html
## Not run:
toxlsx(object = iris, path = tempdir())

## End(Not run)
```

`xls_theme` *Constructor function for xls themes*

Description

This function creates an xls theme for styling exported tables. All its arguments must be ‘openxlsx‘ Style objects.

Usage

```
xls_theme(
  title,
  col_header,
  character,
  footnote1,
  footnote2,
  footnote3,
  mergedcell,
  ...
)
```

Arguments

`title`	Style for the title
`col_header`	Style for the columns header
`character`	Default style for data cells
`footnote1`	Style for footnote1
`footnote2`	Style for footnote2
`footnote3`	Style for footnote3
`mergedcell`	Style for merged cells
`...`	Other (named) custom styles

Value

a named list of class xls_theme, whose elements are ‘openxlsx‘ Style objects.

See Also

`xls_theme_plain()`, `xls_theme_default()`

Examples

```
my_theme <- xls_theme(
  title = openxlsx::createStyle(),
  col_header = openxlsx::createStyle(),
  character = openxlsx::createStyle(),
  footnote1 = openxlsx::createStyle(),
  footnote2 = openxlsx::createStyle(),
  footnote3 = openxlsx::createStyle(),
  mergedcell = openxlsx::createStyle()
)

## Not run:
toxlsx(object = iris, path = tempdir(), theme = my_theme)

## End(Not run)
```

`xls_theme_default` *Constructor function for the default xls theme*

Description

This function is a wrapper around [xls_theme()] that creates an xls theme for styling exported tables. It defines a theme whith sensible default formatting values. It also defines custom styles for "number", "decimal" and "percent column types. All its arguments must be ‘openxlsx‘ Style objects.

Usage

```
xls_theme_default(
  title = openxlsx::createStyle(fontSize = 16, textDecoration = "bold"),
  footnote1 = openxlsx::createStyle(fontSize = 12),
  footnote2 = openxlsx::createStyle(fontSize = 12),
  footnote3 = openxlsx::createStyle(fontSize = 12),
  col_header = openxlsx::createStyle(fontSize = 12, textDecoration = "bold", border =
    c("top", "bottom", "left", "right"), borderStyle = "thin", wrapText = TRUE, halign =
    "center"),
  character = openxlsx::createStyle(fontSize = 12, border = c("top", "bottom", "left",
    "right"), borderStyle = "thin"),
  number = openxlsx::createStyle(fontSize = 12, numFmt = "### ### ### ##0", border =
    c("top", "bottom", "left", "right"), borderStyle = "thin"),
  decimal = openxlsx::createStyle(fontSize = 12, numFmt = "### ### ### ##0.0", border =
    c("top", "bottom", "left", "right"), borderStyle = "thin"),
  percent = openxlsx::createStyle(fontSize = 12, numFmt = "#0.0", border = c("top",
    "bottom", "left", "right"), borderStyle = "thin", halign = "center"),
  mergedcell = openxlsx::createStyle(fontSize = 12, border = c("top", "bottom", "left",
    "right"), borderStyle = "thin", wrapText = TRUE, valign = "center", halign =
    "center"),
  ...
)
```

Arguments

`title`	Style for the title
`footnote1`	Style for footnote1
`footnote2`	Style for footnote2
`footnote3`	Style for footnote3
`col_header`	Style for the columns header
`character`	Default style for data cells
`number`	Style for columns in number format
`decimal`	Style for columns in decimal format
`percent`	Style for columns in percent format
`mergedcell`	Style for merged cells
`...`	Other (named) custom styles

Value

a named list of class xls_theme, whose elements are ‘openxlsx‘ Style objects.

See Also

`xls_theme()`, `xls_theme_plain()`

Examples

```
# default theme
xls_theme_default()

# default theme with title in italic
my_theme <- xls_theme_default(title = openxlsx::createStyle(textDecoration = "italic"))

## Not run:
toxlsx(object = iris, path = tempdir(), theme = my_theme)

## End(Not run)
```

xls_theme_plain *Constructor function for a plain xls theme*

Description

This function is a wrapper around [xls_theme()] that creates an xls theme for styling exported tables. It defines a simple theme whith no special formatting. All its arguments must be ‘openxlsx‘ Style objects.

Usage

```
xls_theme_plain(
  title = openxlsx::createStyle(),
  col_header = openxlsx::createStyle(),
  character = openxlsx::createStyle(),
  footnote1 = openxlsx::createStyle(),
  footnote2 = openxlsx::createStyle(),
  footnote3 = openxlsx::createStyle(),
  mergedcell = openxlsx::createStyle(),
  ...
)
```

Arguments

`title`	Style for the title
`col_header`	Style for the columns header
`character`	Default style for data cells
`footnote1`	Style for footnote1
`footnote2`	Style for footnote2
`footnote3`	Style for footnote3
`mergedcell`	Style for merged cells
`...`	Other (named) custom styles

Value

a named list of class xls_theme, whose elements are ‘openxlsx‘ Style objects.

See Also

xls_theme(), xls_theme_default()

Examples

```
# plain theme
xls_theme_plain()

# plain theme with title in bold
my_theme <- xls_theme_plain(title = openxlsx::createStyle(textDecoration = "bold"))

## Not run:
toxlsx(object = iris, path = tempdir(), theme = my_theme)

## End(Not run)
```

Index