Package 'xlsimple'

August 1, 2020

Type Package

Title 'XLConnect' Wrapper

Description Provides a simple wrapper for some 'XLConnect' functions. 'XLConnect' is a package that allows for reading, writing, and manipulating 'Microsoft Excel' files. This package, 'xlsimple', adds some documentation and pre-defined formatting to the outputted 'Excel' file. Individual sheets can include a description on the first row to remind user what is in the data set. Auto filters and freeze rows are turned on. A brief readme file is created that provides a summary listing of the created sheets and, where provided, the description.

Maintainer Erik W. Leppo <Erik.Leppo@tetratech.com>

Version 1.0.5

Date 2020-07-30

Depends R (>= 3.2.0)

Imports XLConnect

Suggests knitr

SystemRequirements Java (>= 8, <= 11)

License GPL-3

Encoding UTF-8

LazyData TRUE

RoxygenNote 7.1.1

NeedsCompilation no

Author Jon Harcum [aut],
Erik W. Leppo [aut, cre]

Repository CRAN

Date/Publication 2020-08-01 00:50:03 UTC

R topics documented:

`.addXLreadme` *Add a readme sheet to the 'Excel' workbook*

Description

Add a readme sheet to the 'Excel' workbook

Usage

```
.addXLreadme(wbList = XL.wb)
```

Arguments

`wbList`	list with workbook and default cell styles (i.e., output from getXLsettings)

Value

list with workbook and default cell styles

`.findFile` *Find Recent File Information*

Description

Find recent file information based on folder and file name (allows for wildcard structure)

Usage

```
.findFile(folder = ".", file = "*.*", n = 1, fileNameOnly = TRUE)
```

Arguments

`folder`	folder (i.e., directory to look in, can use relative path)
`file`	file (can use wildcards, e.g., "*.csv")
`n`	number of files to return (e.g., value of 1 returns most recent file, value of 'all' returns all files)
`fileNameOnly`	logical field, if TRUE only return file name

Details

This function is used to search a selected directory for information about the most recently modified files in that directory. The default setting searches the current working directory. Relative directory addresses can be used. The default settings returns the name of the most recently modified file. Employing wildcards in the file argument can narrow the file search, e.g., "*.csv" will only return comma delimited files.

The default setting for the argument, n (of 1), will only return a single file. This value can be increased to any number (2, 3,...) to change the maximum number of files returned; or the argument can be set to 'all' to return all files. Setting the argument, fileNameOnly, to FALSE will result in returning additional file meta data related to file size, modified date/time and create date/time.

The results are in descending order of modified date/time.

Value

returns file name as a character string

Examples

```
# name of most recently modified file
.findFile()          # current directory
.findFile("..")      # one directory up
.findFile(tempdir()) # temp directory

# list of files and common attributes in temp directory
.findFile(folder=tempdir(), file="*.*", n=2, fileNameOnly=FALSE)    #two most recent files
.findFile(folder=tempdir(), file="*.*", n="all", fileNameOnly=FALSE)  #all files
```

---

addXLsheetStd *Add a sheet to the 'Excel' workbook*

Description

Add a sheet to the 'Excel' workbook

Usage

```
addXLsheetStd(wbList = XL.wb, df = NA, sheetName = NA, descrip = NA)
```

Arguments

wbList	list with workbook and default cell styles (i.e., output from getXLsettings)
df	data frame to output to sheet
sheetName	sheet name
descrip	description of sheet

Details

The sheetName is the name that will be used for the sheet. When sheetName is not specified, the name of the df will be used. The descrip is a character string that will be placed into cell A1 of the sheet and is best used to provide a brief description of the sheet; and the dataframe will begin at cell A3. If descrip is not specified, then the dataframe will begin at cell A1. The outputted table will have filters turned on and table headers and first column frozen.

Value

list with workbook and default cell styles

Examples

```
XL.wb <- getXLsettings()
XL.wb <- addXLsheetStd(XL.wb, mtcars)
XL.wb <- addXLsheetStd(XL.wb, mtcars, "mtcars1")
XL.wb <- addXLsheetStd(XL.wb, mtcars, "mtcars2", "Standard mtcars data frame")
XL.wb$pName <- "ProjName" # optional, blank if not included
XL.wb$pDesc <- "ProjDesc" # optional, blank if not included
saveXLworkbook(XL.wb, file.path(tempdir(), 'myXLfile.xlsx'), timeStamp=FALSE, clean=FALSE)
saveXLworkbook(XL.wb, file.path(tempdir(), 'myXLfile.xlsx'), timeStamp=TRUE, clean=FALSE)
saveXLworkbook(XL.wb, file.path(tempdir(), 'myXLfile.xlsx'), timeStamp=TRUE, clean=TRUE)
saveXLworkbook(XL.wb, file.path(tempdir(), 'myXLfile.xlsx'))
```

---

getXLsettings *Get Default 'Excel' cell styles*

Description

Get Default 'Excel' cell styles

Usage

```
getXLsettings(fname = NA)
```

Arguments

fname	'Excel' file name

Details

If no 'Excel' file is specified then the built in cell styles are used. User-supplied 'Excel' file would need to have the following cell styles specified: xl.descrip, xl.normal, xl.header, xl.header.wrap, xl.hyperlink.

Value

list with workbook and default cell styles

Examples

```
XL.wb <- getXLsettings()
XL.wb <- addXLsheetStd(XL.wb, mtcars)
XL.wb <- addXLsheetStd(XL.wb, mtcars, "mtcars1")
XL.wb <- addXLsheetStd(XL.wb, mtcars, "mtcars2", "Standard mtcars data frame")
XL.wb$pName <- "ProjName" # optional, blank if not included
XL.wb$pDesc <- "ProjDesc" # optional, blank if not included
saveXLworkbook(XL.wb, file.path(tempdir(), 'myXLfile.xlsx'), timeStamp=FALSE, clean=FALSE)
saveXLworkbook(XL.wb, file.path(tempdir(), 'myXLfile.xlsx'), timeStamp=TRUE, clean=FALSE)
saveXLworkbook(XL.wb, file.path(tempdir(), 'myXLfile.xlsx'), timeStamp=TRUE, clean=TRUE)
saveXLworkbook(XL.wb, file.path(tempdir(), 'myXLfile.xlsx'))
```

---

readXLworkbook *Load a full 'MS Excel' File*

Description

Given a 'MS Excel' File it loads the full workbook creating as many variables as sheets exists in the workbook.

Usage

```
readXLworkbook(
  filename,
  environment = parent.frame(),
  verbose = TRUE,
  free.warnings = TRUE
)
```

Arguments

filename	The path and the name to the' MS Excel' File.
environment	Environment were the new variables will be located. By default is parent environment.
verbose	If set to TRUE useful messages are shown.
free.warnings	If set to TRUE it shows any warnings result of loading the content of the book's sheets.

Details

This function was developed by Carles Hernandez-Ferrer (carleshf) in the package "loadxls" hosted on GitHub. As of 2020-07-20 the repo no longer exists. The blog post about the package is below.

https://carleshf87.wordpress.com/2016/01/24/loadxls-r-package/

saveXLworkbook *Save 'Excel' Workbook to disk*

Description

Save 'Excel' Workbook to disk

Usage

```
saveXLworkbook(wbList, fname = "xl.Out.xlsx", timeStamp = FALSE, clean = TRUE)
```

Arguments

wbList	list with workbook and default cell styles (i.e., output from getXLsettings)
fname	'Excel' file name
timeStamp	Logical field to include date/time stamp in file name (TRUE [default]).
clean	Logical field indicating whether to remove original sheets in workbook

Value

n/a

Examples

```
XL.wb <- getXLsettings()
XL.wb <- addXLsheetStd(XL.wb, mtcars)
XL.wb <- addXLsheetStd(XL.wb, mtcars, "mtcars1")
XL.wb <- addXLsheetStd(XL.wb, mtcars, "mtcars2", "Standard mtcars data frame")
XL.wb$pName <- "ProjName" # optional, blank if not included
XL.wb$pDesc <- "ProjDesc" # optional, blank if not included
saveXLworkbook(XL.wb, file.path(tempdir(), 'myXLfile.xlsx'), timeStamp=FALSE, clean=FALSE)
saveXLworkbook(XL.wb, file.path(tempdir(), 'myXLfile.xlsx'), timeStamp=TRUE, clean=FALSE)
saveXLworkbook(XL.wb, file.path(tempdir(), 'myXLfile.xlsx'), timeStamp=TRUE, clean=TRUE)
saveXLworkbook(XL.wb, file.path(tempdir(), 'myXLfile.xlsx'))
```

Index