Package 'thinkr'

June 21, 2018

Type Package

Title Tools for Cleaning Up Messy Files

Version 0.13

Date 2018-06-19

Description
Some tools for cleaning up messy 'Excel' files to be suitable for R. People who have been working with 'Excel' for years built more or less complicated sheets with names, characters, formats that are not homogeneous. To be able to use them in R nowadays, we built a set of functions that will avoid the majority of importation problems and keep all the data at best.

License GPL-3

Imports assertthat, devtools, dplyr, ggplot2, lazyeval, lubridate, magrittr, officer, stats, stringi, stringr, tidyr, utils, rvg

Suggests knitr, rmarkdown, covr, testthat

LazyData TRUE

RoxygenNote 6.0.1

Depends R (>= 3.1)

NeedsCompilation no

Author Vincent Guyader [aut, cre]

Maintainer Vincent Guyader <vincent@thinkr.fr>

Repository CRAN

Date/Publication 2018-06-21 07:41:47 UTC

R topics documented:

`all_ggplot_to_pptx` *Save all ggplot in a pptx*

Description

Save all ggplot in a pptx

Usage

```
all_ggplot_to_pptx(out = "tous_les_graphs.pptx", open = TRUE, png = TRUE,
  folder = "dessin", global = TRUE)
```

Arguments

`out`	output file name
`open`	booleen open file after creation
`png`	booleen also save as png
`folder`	png's folder
`global`	booleen use .GlobalEnv

Examples

```
## Not run:
all_ggplot_to_pptx()
## End(Not run)
```

as_mon_numeric *transform a vector into numeric*

Description

transform a vector into numeric

Usage

```
as_mon_numeric(vec)
```

Arguments

vec	a vector

Value

a numeric vector

clean_levels *Clean levels label*

Description

Clean levels label

Usage

```
clean_levels(vec, verbose = FALSE, translit = FALSE, punct = FALSE)
```

Arguments

vec	a factor
verbose	booleen is the function verbose
translit	booleen remove non ascii character
punct	booleen do you remove punctuation

clean_names *clean_names*

Description

clean_names

Usage

```
clean_names(dataset, verbose = FALSE, translit = TRUE)
```

Arguments

dataset	a dataframe
verbose	logical
translit	logical remove non ascii character

Value

a dataframe

Examples

```
data(iris)
clean_names(iris)
```

clean_vec *Clean character vector*

Description

Clean character vector

Usage

```
clean_vec(vec, verbose = FALSE, unique = TRUE, keep_number = FALSE,
  translit = TRUE, punct = TRUE)
```

Arguments

vec	character vector to clean
verbose	logical is the function verbose
unique	logical do we have to apply make_unique
keep_number	logical keep number at begining
translit	logical remove non ascii character
punct	logical do you remove punctuation

dput_levels *return R instruction to create levels*

Description

return R instruction to create levels

Usage

```
dput_levels(vec)
```

Arguments

vec	a factor or character vector

Value

a R instruction

Examples

```
dput_levels(iris$Species)
```

excel_col *return all excel column name*

Description

return all excel column name

Usage

```
excel_col()
```

Examples

```
excel_col()
ncol_to_excel(6)
excel_to_ncol('AF')
```

excel_to_ncol *return excel column position number from a column name*

Description

return excel column position number from a column name

Usage

```
excel_to_ncol(col_name)
```

Arguments

col_name	the culum name

Examples

```
excel_to_ncol("BF")
```

find_name *find pattern in name's dataset*

Description

find pattern in name's dataset

Usage

```
find_name(dataset, pattern)
```

Arguments

dataset	a data.frame (or list or anything with names parameter)
pattern	pattern we are looking for

Value

a list with position and value

Examples

```
find_name(iris,"Sepal")
```

from_excel_to_posixt *transform the excel numeric date format into POSIXct*

Description

transform the excel numeric date format into POSIXct

Usage

```
from_excel_to_posixt(vec, origin = "1904-01-01")
```

Arguments

vec	a vector
origin	a date-time object, or something which can be coerced by as.POSIXct(tz = "GMT") to such an object.

gsub2 *like gsub but keep a factor as factor*

Description

like gsub but keep a factor as factor

Usage

```
gsub2(x, ...)
```

Arguments

x	a vector
...	les parametres de la fonction gsub

Value

a vector

is.01 *does this vector only contains 0 and 1*

Description

does this vector only contains 0 and 1

Usage

```
is.01(x)
```

Arguments

x	a vector

Value

a boolean

Examples

```
is.01(c(0,1,0,0,1))
is.01(c(0,1,0,0,5))
```

is.12 *does this vector only contains 1 and 2*

Description

does this vector only contains 1 and 2

Usage

```
is.12(x)
```

Arguments

x	a vector

Value

a boolean

Examples

```
is.12(c(1,1,2,1,2))
is.12(c(1,1,2,1,5))
```

---

is_full_figures *Predicate for charater vector full of figures*

Description

detects if a character vector is only made with figures. Useful when you

Usage

```
is_full_figures(.)
```

Arguments

.	a vector of character (and eventually NA's)

Value

a boolean

Examples

```
is_full_figures(c(NA,"0","25.3"))
is_full_figures((c(NA,"0","25_3")))
```

---

is_full_na *Predicate for full NA vector*

Description

is_full_na test if the vector is full of NA's

Usage

```
is_full_na(.)
```

Arguments

.	a vector

Value

a vector of boolean

Examples

```
is_full_na(c(NA, NA, NA))
```

---

is_likert *is a factor a likert scale*

Description

is a factor a likert scale

Usage

```
is_likert(vec, lev)
```

Arguments

vec	a factor
lev	le scale

Value

boolean

Examples

```
is_likert(iris$Species,c("setosa","versicolor","virginica"))
is_likert(iris$Species,c("setosa","versicolor","virginica","banana"))
is_likert(iris$Species,c("setosa","versicolor"))
```

---

look_like_a_number *return TRUE if this look like a number*

Description

return TRUE if this look like a number

Usage

```
look_like_a_number(vec)
```

Arguments

vec	a vector

Value

un booleen

---

make_unique *make.unique improvement*

Description

make.unique improvement

Usage

```
make_unique(vec, sep = "_")
```

Arguments

vec	a vector
sep	char separator to use

Value

a vector

Examples

```
make_unique(c("a","a","a","b","a","b","c"))
```

---

ncol_to_excel *return excel column name from a position number*

Description

return excel column name from a position number

Usage

```
ncol_to_excel(n)
```

Arguments

n	the colum position

Examples

```
ncol_to_excel(35)
```

---

peep *peep the pipeline*

Description

peep some data at one step of a pipeline.

Usage

```
peep(data, ..., printer = print, verbose = FALSE)
```

Arguments

data	some data
...	function names or expressions that use . as a placeholder for the data
printer	which function use to print
verbose	TRUE to include what is printed

Value

the input data

Examples

```
if( require(magrittr) ){
  # just symbols
  iris %>% peep(head,tail) %>% summary
  # expressions with .
  iris %>% peep(head(., n=2),tail(., n=3) ) %>% summary
  # or both
  iris %>% peep(head,tail(., n=3) ) %>% summary
  # use verbose to see what happens
  iris %>% peep(head,tail(., n=3), verbose = TRUE) %>% summary
}
```

replace_pattern *Replace pattern everywhere in a data.frame*

Description

Replace pattern everywhere in a data.frame

Usage

```
replace_pattern(dataset, pattern, replacement, exact = FALSE)
```

Arguments

dataset	a data.frame
pattern	Pattern to look for.
replacement	A character of replacements.
exact	a boolean if TRUE the whole value need ton match

Value

a data.frame

Examples

```
library(dplyr)
library(tidyr)
dataset <-
data.frame(a=as.factor(letters)[1:7],b=letters[1:7],c=1:7,stringsAsFactors = FALSE) %>%
unite("fus",a,b,remove=FALSE,sep="")
dataset %>% replace_pattern("a",'"XXX') %>% summary()
```

save_as_csv *export a data.frame to csv*

Description

export a data.frame to csv

Usage

```
save_as_csv(dataset, path, row.names = FALSE, ...)
```

Arguments

dataset	a data.frame
path	the path
row.names	booleen do we have to save the row names
...	other write.csv parameters

Value

file name as character

Examples

```
## Not run:
iris %>% save_as_csv(file.path(tempdir(),'coucou.csv')) %>% browseURL()



## End(Not run)
```

---

set_col_type *set a given coltype to each column in a data.frame*

Description

set a given coltype to each column in a data.frame

Usage

```
set_col_type(dataset, col_type)
```

Arguments

dataset	a data.frame
col_type	a character vector containing the class to apply

Value

a data.frame

---

thinkr *thinkr*

Description

a package with some (useful) tools (data manipulation)

`%ni%` *not in*

Description

not in

Usage

```
x %ni% table
```

Arguments

`x`	vector or NULL: the values to be matched
`table`	the values to be matched against

Examples

```
"a" %ni% letters
"coucou" %ni% letters
```

Index