Package 'colr'

January 3, 2017

Type Package

Title Functions to Select and Rename Data

Version 0.1.900

Date 2017-01-03

Author Louis Chaillet <commerce.chaillet@gmail.com>

Maintainer Louis Chaillet <commerce.chaillet@gmail.com>

Description Powerful functions to select and rename columns in dataframes, lists and numeric types by 'Perl' regular expression. Regular expression ('regex') are a very powerful grammar to match strings, such as column names.

Depends R (>= 3.0.0)

License GPL (>= 2)

LazyData TRUE

RoxygenNote 5.0.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2017-01-03 18:30:54

R topics documented:

cgrep *a function to select columns by 'regex'*

Description

Select columns (or rows) by 'perl' regular expression. See regex{base} for 'regex' documenation. 'regex' is a very powerful grammar to match strings.

Usage

```
cgrep(x, pattern, dim = c("c", "r"))
```

Arguments

`x` A dataframe, list a matrix with column names or a named numeric.

`pattern` A search string

`dim` A character either "c" for selection of columns or "r" for selection of rows, if x is a list this parameter has no meaning

Value

A Dataframe, list or matrix where the column- or row names match the pattern. By default the selecion is by column, unless the dim parameter specifies "r" for rows. If f x is not a dataframe and if a single column or row is selected 'R' will flatten the results to a vector.

See Also

regex{base}

Examples

```
head(cgrep(iris, "^.+$")) # matches all columns that have non-empty column names and thus
#drops all columns with empty names

head(cgrep(iris, "^Petal\\.")) # matches all columns that have  names starting with the
#string "Petal."
head(cgrep(iris, "\\.")) # columns with names that contain a dot
head(cgrep(iris, "\\.[Ww]idth$")) # columns with names ending in the string ".width" or ".Width".
head(cgrep(iris, "^[SP]e.al")) # columns that have names starting with either capital
#'S' or 'P' then an 'e' followed by any character and then 'al'
```

`colrdata` *Dummy data for demonstration of the capacities of 'rcol'*

Description

A dataset containing with columns with malformed dates.

Usage

```
colrdata
```

Format

A data frame with 3 rows and 5 variables:

01-31-1955 dummy

02-15-1980 dummy

06/02/1999 dummy

02 14 01 dummy

03 03-2016 dummy...

Source

my thumb

`csub` *a function to change column names*

Description

a function to change column names

Usage

```
csub(x, pattern, replacement, dim = c("c", "r"), gl = TRUE)
```

Arguments

`x`	A dataframe, list or matrix with column names
`pattern`	A string
`replacement`	A string
`dim`	A character either "c" for selection of columns or "r" for selection of rows, if x is a list this parameter has no meaning
`gl`	A boolean, if TRUE (the default) all occurences in the input strings (row or column names) are replaced, if FALSE only the first occurence in every string is replaced.

Value

A dataframe, list or matrix where the columns or rows are renamed based on the 'Regex' substitution defined in the pattern and the replacement. By default column are renamed, unless the dim parameter specifies "r" for rows. The default is to use gsub (replacing all occurences of pattern in each column name), to replace only the first occurence set gl to FALSE.

For lists the function acts on the highest level of the list and this is irrespective of the gl parameter.

See Also

regex{base}

Examples

```
head(csub(iris, "\\.", "-")) # will change all dots in column names in "-"
head(csub(iris, "[pP]etal", "Beetle"))

# a more complex example showing the power of 'regex'

head(csub(data.frame(WorldPhones),"^([NM](?:\\w{2})?\\.)Amer", "\\1America"))


# this example will also work on other types (the call data.frame is not necessary)
```

Index