

Package ‘textutils’

October 14, 2022

Type Package

Title Utilities for Handling Strings and Text

Version 0.2-1

Date 2021-04-01

Imports utils

Suggests tinytest

Maintainer Enrico Schumann <es@enricoschumann.net>

Description Utilities for handling character vectors that store human-readable text (either plain or with markup, such as HTML or LaTeX). The package provides, in particular, functions that help with the preparation of plain-text reports, e.g. for expanding and aligning strings that form the lines of such reports. The package also provides generic functions for transforming R objects to HTML and to plain text.

License GPL-3

URL <http://enricoschumann.net/R/packages/textutils/>,
<https://github.com/enricoschumann/textutils>

NeedsCompilation no

Author Enrico Schumann [aut, cre] (<<https://orcid.org/0000-0001-7601-6576>>)

Repository CRAN

Date/Publication 2021-04-01 07:20:02 UTC

R topics documented:

textutils-package	2
btable	2
dctable	3
fill_in	4
here	5
HTMLencode	7

latexrule	8
rmp	9
spaces	10
stexp	11
TeXencode	12
TeXunits	13
title_case	14
toHTML	15
toLatex.data.frame	16
toText	17
trim	18
valign	19

Index	21
--------------	-----------

textutils-package	<i>Utilities for Handling Strings and Text</i>
-------------------	--

Description

Utilities for handling character vectors that store human-readable text (either plain or with markup, such as HTML or LaTeX). The package provides, in particular, functions that help with the preparation of plain-text reports, e.g. for expanding and aligning strings that form the lines of such reports. The package also provides generic functions for transforming R objects to HTML and to plain text.

Details

The package comprises a number of functions that help with manipulating character strings.

For more information and a complete list of functions, use `'library(help = "textutils")'`.

Author(s)

NA

Maintainer: Enrico Schumann <es@enricoschumann.net>

btable	<i>Barplot Table</i>
--------	----------------------

Description

Create a LaTeX-table.

Usage

```
btable(x, unit = "cm", before = "", after = "", raise = "0.2ex",
       height = "1ex", ...)
```

Arguments

x	numeric: the numbers for which the barplot is to be created
unit	character: a valid TeX unit
before	character
after	character
raise	character
height	character
...	more arguments

Details

Creates a barplot table.

Value

character

Author(s)

Enrico Schumann

See Also

[toLatex](#), [TeXunits](#)

Examples

```
## see vignette
```

dctable

Dotchart Table

Description

Create a LaTeX-table.

Usage

```
dctable(x, unitlength = "1 cm", width = 5,  
        y.offset = 0.07, circle.size = 0.1, xlim,  
        na.rm = FALSE)
```

Arguments

x	numeric: the numbers for which the barplot is to be created
unitlength	character
width	numeric
y.offset	numeric
circle.size	numeric
xlim	character
na.rm	logical

Details

Creates a dotchart table.

This function is currently very experimental.

Value

character

Author(s)

Enrico Schumann

References

Cleveland, W. S. (1985) *The Elements of Graphing Data*. Wadsworth.

See Also

[toLatex](#), [TeXunits](#)

Examples

```
## see vignette
```

fill_in

Fill In Templates

Description

Light-weight template filling: replace placeholders in a string by values.

Usage

```
fill_in(s, ..., delim = c("{", "}"), replace.NA = TRUE)
```

Arguments

s character
 ... typically name/value pairs. See Examples.
 delim characters
 replace.NA logical: if TRUE, NA values are replaced by the string "NA". May also be a string. See Examples.

Details

A light-weight replacement function.

Value

character

Author(s)

Enrico Schumann

Examples

```
template <- "{1} meets {2}"
fill_in(template, "Peter", "Paul") ## "Peter meets Paul"

template <- "{one} meets {other}"
fill_in(template, one = "Peter", other = "Paul") ## "Peter meets Paul"

## handling missing values
fill_in("{name}: {score}", name = "Peter", score = NA)
## [1] "Peter: NA"

fill_in("{name}: {score}", name = "Peter", score = NA, replace.NA = ".")
## [1] "Peter: ."
```

here

Here Documents

Description

Read lines and convert into appropriate vector or data frame.

Usage

```
here(s, drop = TRUE, guess.type = TRUE, sep = NULL, header = TRUE,
     stringsAsFactors = FALSE, trim = TRUE, ...)
```

Arguments

<code>s</code>	a string
<code>drop</code>	logical: drop empty first and last element
<code>guess.type</code>	logical
<code>sep</code>	NULL or character
<code>header</code>	logical
<code>stringsAsFactors</code>	logical
<code>trim</code>	logical: trim whitespace?
<code>...</code>	named arguments to be passed to read.table

Details

Experimental. (Notably, the function's name may change.)

The function reads a (typically multi-line) string and treats each line as one element of a vector or, if `sep` is specified, a `data.frame`.

If `sep` is not specified, here calls [type.convert](#) on the input `s`.

If `sep` is specified, the input `s` is fed to [read.table](#). Additional arguments may be passed through
....

Value

a vector or, if `sep` is specified, a `data.frame`

Author(s)

Enrico Schumann

References

http://rosettacode.org/wiki/Here_document

(note that R supports multi-line strings, so in a way it has built-in support for here documents as defined on that website)

See Also

[type.convert](#)

Examples

```
## numbers
here("
1
2
3
4
```

```

")

## character
here("
Al
Bob
Carl
David
")

## data frame
here("
letter, number
      x,      1
      y,      2
      z,      3",
sep = ",")

```

HTMLencode

Decode and Encode HTML Entities

Description

Decode and encode HTML entities.

Usage

```

HTMLdecode(x, named = TRUE, hex = TRUE, decimal = TRUE)
HTMLencode(x, use.iconv = FALSE, encode.only = NULL)

```

Arguments

<code>x</code>	a string (character vector of length one)
<code>use.iconv</code>	logical. Should conversion via <code>iconv</code> be tried from native encoding to UTF-8?
<code>named</code>	logical: replace named character references?
<code>hex</code>	logical: replace hexadecimal character references?
<code>decimal</code>	logical: replace decimal character references?
<code>encode.only</code>	character

Details

`HTMLdecode` replaces named, hexadecimal and decimal character references as defined by HTML5 (see References) with characters. The resulting character vector is marked as UTF-8 (see [Encoding](#)).

`HTMLencode` replaces UTF-8-encoded substrings with HTML5 named entities (a.k.a. “named character references”). A semicolon ‘;’ will not be replaced by the entity ‘;’. Other than that, however, `HTMLencode` is quite thorough in its job: it will replace all characters for which named entities exists, even ‘,’ and or ‘?’. You can restrict the characters to be replaced by specifying `encode.only`.

Value

character

Author(s)

Enrico Schumann

References

<https://www.w3.org/TR/html5/syntax.html#named-character-references>

<https://html.spec.whatwg.org/multipage/syntax.html#character-references>

See Also

[TeXencode](#)

Examples

```
HTMLdecode(c("Max & Moritz", "4 &lt; 9"))
## [1] "Max & Moritz" "4 < 9"
```

```
HTMLencode(c("Max & Moritz", "4 < 9"))
## [1] "Max & Moritz" "4 &LT; 9"
```

```
HTMLencode("Max, Moritz & more")
## [1] "Max&comma; Moritz & more"
HTMLencode("Max, Moritz & more", encode.only = c("&", "<", ">"))
## [1] "Max, Moritz & more"
```

latexrule

LaTeX Rule.

Description

Create a LaTeX-rule, including colours.

Usage

```
latexrule(x, y, col = NULL, x.unit = "cm", y.unit = "cm", noindent = FALSE)
```

Arguments

x	numeric
y	numeric
col	character
x.unit	character
y.unit	character
noindent	logical

Details

Experimental. Create LaTeX code that produces rules.

Value

character

Author(s)

Enrico Schumann

Examples

```
## see vignette
```

rmp

Remove Repeated Pattern

Description

Remove a repeated pattern in a character vector.

Usage

```
rmp(s, pattern, ...)
```

Arguments

s	a character vector
pattern	a regular expression
...	arguments passed to grep

Details

rmp removes a repeated pattern in a character vector (e.g. repeated blank lines).

Value

a character vector

Author(s)

Enrico Schumann

See Also

[strwrap](#), [format](#)

Examples

```
## remove repeated blanks from vector
s <- c("* Header", "", " ", "", "** Subheader")
rmp(s, "^ *$")
```

spaces

Create Vectors of White Space

Description

Create character vectors of white space.

Usage

```
spaces(n)
```

Arguments

n integer

Details

The function creates a character vector of white-space strings. Such vectors are useful, for instance, for padding character vectors.

Value

character

Author(s)

Enrico Schumann

See Also

[strep](#)

Examples

```
spaces(0:3)
```

strexp *Expand String to Fixed Width*

Description

Expand strings to a fixed 'length' (in the sense of [nchar](#)).

Usage

```
strexp(s, after, width, fill = " ", at)
```

Arguments

s	a character vector
after	character: a pattern, to be passed to regexpr
width	integer
fill	character
at	integer

Details

strexp inserts blanks into the elements of a character vector such that all elements have the same width (i.e. [nchar](#)). Note that it will (currently) not contract a string, only expand it.

Value

a character vector

Author(s)

Enrico Schumann

See Also

[strwrap](#), [format](#)

Examples

```
## expand to width 10, but keep two initial blanks
s <- c(" A 1", " B    2")
strexp(s, after = " +[^ ]+ +", width = 10)
```

TeXencode

Encode Special Characters for TeX/LaTeX

Description

Encode special characters for TeX/LaTeX.

Usage

TeXencode(s)

Arguments

s character

Details

Probably incomplet

Value

numeric

Author(s)

Enrico Schumann

References

Donald E. Knuth. *The TeXbook*. Addison Wesley, 1986 (with corrections made in 1996).

Leslie Lamport. *LaTeX: A Document Preparation System*. Addison Wesley, 1994.

Examples

```
TeXencode("Peter & Paul")  
## [1] "Peter \& Paul"
```

Description

Translates units of measurement known to TeX and LaTeX.

Usage

```
TeXunits(from, to, from.unit = NULL)
```

Arguments

from	Typically character, such as "1in". When numeric, from.unit needs to be specified.
to	character
from.unit	character

Details

Available units are centimetre (cm), inch (in), point (pt), pica (pc), big point(bp), millimetre (mm), Didot points (dd) and Cicero (cc).

See Chapter 10 of the TeXbook for details.

Value

numeric

Author(s)

Enrico Schumann

References

Donald E. Knuth. *The TeXbook*. Addison Wesley, 1986 (with corrections made in 1996).

Examples

```
TeXunits("1in",  
         c("in", "mm", "pt", "in"))  
TeXunits(c("1in", "2in"),  
         "cm")
```

title_case	<i>Remove Leading and Trailing White Space</i>
------------	--

Description

Remove leading and/or trailing white space from character vectors.

Usage

```
title_case(s, strict = FALSE, ignore = NULL)
```

Arguments

s	a character vector
strict	logical: if TRUE, only the first letter of each word is uppercase
ignore	character

Details

Set string in title case.

Value

a character vector

Author(s)

Enrico Schumann

See Also

[tolower](#), [toupper](#).

Examples

```
title_case("text mining")
```

`toHTML`*Convert R Objects to HTML*

Description

Convert an R object to an HTML snippet.

Usage

```
toHTML(x, ...)  
  
## S3 method for class 'data.frame'  
toHTML(x, ...,  
        row.names = FALSE,  
        col.names = TRUE,  
        class.handlers = list(),  
        col.handlers = list())
```

Arguments

<code>x</code>	an object
<code>...</code>	arguments passed to methods
<code>row.names</code>	logical
<code>col.names</code>	logical
<code>class.handlers</code>	a list of named functions
<code>col.handlers</code>	a list of named functions

Details

There exists `toHTML` methods in several packages, e.g. in **tools** or **XML**. Package **R2HTML** has a `HTML` generic.

The ‘semantics’ of this function may differ from other implementations: the function is expected to take an arbitrary R object and return an HTML snippet that can be placed in reports (i.e. in the same spirit as `toLatex`). By contrast, the purpose of `toHTML` in **tools** is to provide a whole HTML document.

The `data.frame` method has two `handlers` arguments: these may store helper functions for formatting columns, either of a specific name (`col.handlers`) or of a specific class (`class.handlers`). The functions in `col.handlers` are applied first; and the affected columns are not touched by `class.handlers`. See Examples.

Value

a character vector

Author(s)

Enrico Schumann

See Also[toLatex](#)**Examples**

```
x <- data.frame(a = 1:3, b = rnorm(3))
cat(toHTML(x,
  col.handlers = list(b = function(x) round(x, 1)),
  class.handlers = list(integer = function(x) 100*x)))

## [ pretty-printed... ]
## <tr> <th>a</th> <th>b</th> </tr>
## <tr> <td>100</td><td>-2.3</td> </tr>
## <tr> <td>200</td><td>-0.1</td> </tr>
## <tr> <td>300</td><td>-2.8</td> </tr>
```

toLatex.data.frame *Convert Data Frame to LaTeX*

Description

Convert data frames to character vector in LaTeX markup.

Usage

```
## S3 method for class 'data.frame'
toLatex(object, row.names = FALSE,
  col.handlers = list(), class.handlers = list(),
  eol = "\\", ...)
```

Arguments

object	a data.frame
row.names	include the row names as the first column
col.handlers	a list of named functions
class.handlers	a list of named functions
eol	character: the line ending; may be a vector of length greater than one
...	other arguments

Details

A method for [toLatex](#) that converts data frames into LaTeX markup. Any formatting to be applied must be specified as a function and passed with `col.handlers` and `class.handlers`.

`col.handlers` take precedent over `class.handlers`.

Value

character

Author(s)

Enrico Schumann

See Also

[toLatex](#)

Examples

```
df <- data.frame(letter = letters[1:5],
                 number = runif(5),
                 stringsAsFactors = FALSE)
toLatex(df,
        col.handlers = list(letter = toupper),
        class.handlers = list(numeric = function(x) format(x, digits = 4)),
        eol = "\\[1ex]")
cat(toLatex(df,
           col.handlers = list(letter = toupper),
           class.handlers = list(numeric = function(x) format(x, digits = 4)),
           eol = "\\[1ex]"), sep = "\n")
```

toText

Convert Objects to (Plain) Text

Description

Converts an R object into a text representation.

Usage

```
toText(x, ...)
```

```
## Default S3 method:
toText(x, ...)
```

Arguments

x an object
... arguments passed to methods

Details

A generic function. Method are expected to coerce a given object to lines of human-readable text that can be used, for instance, for reports. The purpose of `toText` is **not** to store data in a form that can be read and understood by R; for that, see [dput](#) or [dump](#).

The `print` method is essentially equivalent to `cat(x, sep = "\n")`.

There is no restriction on encoding, so plain text does not necessarily mean ASCII. But current methods do not map into markup-representations.

Value

A character vector (lines of text), possibly with a class attribute `text`.

Author(s)

Enrico Schumann

See Also

[toLatex](#), [toHTML](#)

Examples

```
toText(c("a", "b", "c"))  
cat(toHTML(toText(c("a", "b", "c"))))
```

trim

Remove Leading and Trailing White Space

Description

Remove leading and/or trailing white space from character vectors.

Usage

```
trim(s, leading = TRUE, trailing = TRUE, perl = TRUE, ...)
```

Arguments

s	a character vector
leading	logical
trailing	logical
perl	logical
...	arguments passed to gsub

Details

`trim` removes leading and trailing space, which is defined through the (Perl) regular expression `\s`. The base package has a function [trimws](#) these days, so you may not actually need the function (any more).

Value

a character vector

Author(s)

Enrico Schumann

See Also

[trimws](#), [gsub](#), [strtrim](#)

Examples

```
s <- c("\t 2 2\n \t", "  ab ")
trim(s)
```

valign	<i>Vertically Align Strings</i>
--------	---------------------------------

Description

Vertically align character vectors.

Usage

```
valign(s, align = "|", insert.at = "<>", replace = TRUE, fixed = TRUE)
```

Arguments

s	a character vector
align	a regular expression
insert.at	a regular expression
replace	logical
fixed	logical

Details

The function expands the elements of a character vector in such a way that the elements are vertically aligned, which can be handy when generating reports. See Examples.

Value

a character vector

Author(s)

Enrico Schumann

See Also

[strwrap](#), [format](#)

Examples

```
s <- c("Player 1 <>| 100",
      "another player <>| 999999")

cat(paste(s, collapse = "\n"))
## Player 1 <>| 100
## another player <>| 999999

cat(paste(valign(s), collapse = "\n"))
## Player 1      100
## another player 999999
```

Index

- * **package**
 - textutils-package, 2
- btable, 2
- data.frame, 6, 15, 16
- dctable, 3
- dput, 18
- dump, 18
- Encoding, 7
- fill_in, 4
- format, 9, 11, 20
- grep, 9
- gsub, 19
- here, 5
- HTMLdecode (HTMLencode), 7
- HTMLencode, 7
- iconv, 7
- latexrule, 8
- NA, 5
- nchar, 11
- read.table, 6
- regexpr, 11
- rmp, 9
- spaces, 10
- strexpr, 10, 11
- strtrim, 19
- strwrap, 9, 11, 20
- TeXencode, 8, 12
- textutils (textutils-package), 2
- textutils-package, 2
- TeXunits, 3, 4, 13
- title_case, 14
- toHTML, 15, 15, 18
- toLatex, 3, 4, 15–18
- toLatex.data.frame, 16
- tolower, 14
- toText, 17
- toupper, 14
- trim, 18
- trimws, 19
- type.convert, 6
- valign, 19