

Package ‘toolbox’

July 22, 2025

Type Package

Title List, String, and Meta Programming Utility Functions

Version 0.1.1

Author Timothy Conwell

Maintainer Timothy Conwell <timconwell@gmail.com>

Description Includes functions for mapping named lists to function arguments, random strings, pasting and combining rows together across columns, etc.

License GPL (>= 2)

Encoding UTF-8

Imports parallel

RoxygenNote 7.2.0

NeedsCompilation no

Repository CRAN

Date/Publication 2022-09-21 20:10:02 UTC

Contents

argNames	2
argumentNamedList	2
castDateString	3
castLogical	3
castNumeric	4
combineCols	4
consolidateList	5
do.call2	5
doubleQuoteText	6
isNULLorNA	7
jsonStr	7
listExtract	8
namesToString	8
pasteCols	9
pastePaths	10

quoteText	10
sampleStr	11

Index	12
-------	----

argNames	Get the names of the arguments to a function
----------	--

Description

Get the names of the arguments to a function

Usage

argNames(x)

Arguments

x A function or string naming a function.

Value

A vector of the names of the arguments to a function.

Examples

argNames("readLines")

argumentNamedList	Create a named list of length 1 using a name stored in a variable as the name.
-------------------	--

Description

Create a named list of length 1 using a name stored in a variable as the name.

Usage

argumentNamedList(name, x)

Arguments

name The name for the item in the list.
x The item to put in the list.

Value

A named list.

Examples

```
argumentNamedList("test_name", 1)
```

castDateString	<i>Format a date string as " from a SQL database to a format compatible with a HTML date input value.</i>
----------------	---

Description

Format a date string as " from a SQL database to a format compatible with a HTML date input value.

Usage

```
castDateString(x)
```

Arguments

x	A string.
---	-----------

Value

A string, formatted YYYY-MM-DD.

Examples

```
castDateString(Sys.time())
```

castLogical	<i>Convert strings to logical.</i>
-------------	------------------------------------

Description

Convert strings to logical.

Usage

```
castLogical(x)
```

Arguments

x	A string.
---	-----------

Value

A string, converted to logical.

Examples

```
castLogical("1")
```

castNumeric	<i>Convert strings to numeric if possible, otherwise remains as is.</i>
-------------	---

Description

Convert strings to numeric if possible, otherwise remains as is.

Usage

```
castNumeric(x)
```

Arguments

x	A string.
---	-----------

Value

A string, converted to numeric if possible.

Examples

```
castNumeric("100")
```

combineCols	<i>Combine columns of a list/data frame into a list by row</i>
-------------	--

Description

Combine columns of a list/data frame into a list by row

Usage

```
combineCols(x, cols = NULL, by_name = FALSE, parallel = FALSE, cores = 1)
```

Arguments

x	A list or data frame.
cols	An optional vector of column positions or names to combine together. If passing column names, set by_name to TRUE. The order of items in cols determines the order of the combined result.
by_name	Boolean, if TRUE, it quotes the items in cols to properly index the list by name (x[[1]] vs x[["col_a"]]).
parallel	Boolean, if TRUE, attempts to use mclapply.
cores	An integer, the number of cores to use if parallel is TRUE.

Value

A list of the values in each column combined together for each row.

Examples

```
combineCols(list("x" = c(1, 2, 3), "y" = c("a", "b", "c")))
```

consolidateList	<i>Group items of a list by name</i>
-----------------	--------------------------------------

Description

Group items of a list by name

Usage

```
consolidateList(x)
```

Arguments

x A named list, likely with names repeating for different positions.

Value

A list with items consolidated by name.

Examples

```
consolidateList(list("col1" = "Test", "col2" = "Hello", "col1" = "Repeated Name"))
```

do.call2	<i>Filters the argument list to match the arguments in what and then calls do.call.</i>
----------	---

Description

Filters the argument list to match the arguments in what and then calls do.call.

Usage

```
do.call2(what, args, quote = FALSE, envir = parent.frame())
```

Arguments

what	See do.call.
args	Argument list, gets filtered to match arguments of what. See do.call.
quote	See do.call.
envir	See do.call.

Value

See do.call.

See Also

do.call

Examples

```
do.call2(intersect, list(x = c(1, 2, 3), y = c(2)))
```

doubleQuoteText	<i>Add double quotes to strings.</i>
-----------------	--------------------------------------

Description

Add double quotes to strings.

Usage

```
doubleQuoteText(
  x,
  char_only = TRUE,
  excluded_chars = c("NULL"),
  null_or_na_as_NULL = TRUE
)
```

Arguments

x	A string.
char_only	TRUE/FALSE, if TRUE, adds quotes only if is.character(x) is TRUE.
excluded_chars	A character vector, will not add quotes if a value is in excluded_chars.
null_or_na_as_NULL	TRUE/FALSE, if TRUE, NULL and NA values are replaced with the string "NULL".

Value

A string, with double quotes added.

Examples

```
doubleQuoteText("Sample quotes.")
```

isNULLorNA	<i>Checks if x is NULL or NA</i>
------------	----------------------------------

Description

Checks if x is NULL or NA

Usage

```
isNULLorNA(x)
```

Arguments

x	A object.
---	-----------

Value

TRUE/FALSE.

Examples

```
isNULLorNA(NULL)
```

jsonStr	<i>Format data as a JSON object (like this: "x": "120").</i>
---------	--

Description

Format data as a JSON object (like this: "x": "120").

Usage

```
jsonStr(name, val)
```

Arguments

name	A string, the name of the JSON entry
val	A string, the value to associate with the JSON entry.

Value

A string, data formatted as a JSON object.

Examples

```
jsonStr(name = "var1", val = "Blue")
```

listExtract	<i>Extract the values from each entry in a list of vectors at a specific index</i>
-------------	--

Description

Extract the values from each entry in a list of vectors at a specific index

Usage

```
listExtract(x, pos)
```

Arguments

x	A list, each item of the list should have equal length.
pos	A integer, the position to extract from each entry in the list.

Value

A list.

Examples

```
listExtract(list(col1 = c(1, 2, 3, 4, 5), col2 = c("a", "b", "c", "d", "e")), 3)
```

namesToString	<i>Pastes the names of a object into a string, optionally quoting the names.</i>
---------------	--

Description

Pastes the names of a object into a string, optionally quoting the names.

Usage

```
namesToString(x, collapse = ",", quote = FALSE)
```

Arguments

x	A named object (vector, list, data.frame)
collapse	A string to separate the collapsed names.
quote	TRUE/FALSE, if TRUE, adds quotes to the names.

Value

A string.

Examples

```
namesToString(c("test" = 1, "this" = 2))
```

pasteCols

Paste together columns of a list/data frame

Description

Paste together columns of a list/data frame

Usage

```
pasteCols(
  x,
  sep = " ",
  collapse = NULL,
  use_paste0 = FALSE,
  cols = NULL,
  by_name = FALSE
)
```

Arguments

x	A list or data frame.
sep	A character sting to separate the terms.
collapse	An optional character string to separate the results.
use_paste0	Boolean, if TRUE, will call paste0 instead of paste.
cols	An optional vector of column positions or names to paste together. If passing column names, set by_name to TRUE. The order of items in cols determines the order of the paste result.
by_name	Boolean, if TRUE, it quotes the items in cols to properly index the list by name (x[[1]] vs x[["col_a"]]).

Value

A string with the values in each column pasted together.

Examples

```
pasteCols(list("x" = c(1, 2, 3), "y" = c("a", "b", "c")))
```

pastePaths	<i>Paste parts of file paths/urls separated with single forward-slashes</i>
------------	---

Description

Paste parts of file paths/urls separated with single forward-slashes

Usage

```
pastePaths(...)
```

Arguments

... Text strings to combine into a file path

Value

A string.

Examples

```
pastePaths("/home/", "/files")
```

quoteText	<i>Add single quotes to strings, useful for converting R strings into SQL formatted strings.</i>
-----------	--

Description

Add single quotes to strings, useful for converting R strings into SQL formatted strings.

Usage

```
quoteText(
  x,
  char_only = TRUE,
  excluded_chars = c("NULL"),
  null_or_na_as_NULL = TRUE
)
```

Arguments

x A string.

char_only TRUE/FALSE, if TRUE, adds quotes only if is.character(x) is TRUE.

excluded_chars A character vector, will not add quotes if a value is in excluded_chars.

null_or_na_as_NULL TRUE/FALSE, if TRUE, NULL and NA values are replaced with the string "NULL".

Value

A string, with single quotes added to match PostgreSQL string formatting.

Examples

```
quoteText("Sample quotes.")
```

sampleStr*Generates (pseudo)random strings of the specified char length*

Description

Generates (pseudo)random strings of the specified char length

Usage

```
sampleStr(n_char, sample_chars = c(letters, LETTERS, 0:9))
```

Arguments

n_char	A integer, the number of chars to include in the output string.
sample_chars	A vector of characters to sample from. Includes the lowercase and uppercase English alphabet and 0-9 by default.

Value

A string.

Examples

```
sampleStr(10)
```

Index

argNames, [2](#)
argumentNamedList, [2](#)

castDateString, [3](#)
castLogical, [3](#)
castNumeric, [4](#)
combineCols, [4](#)
consolidateList, [5](#)

do.call2, [5](#)
doubleQuoteText, [6](#)

isNULLorNA, [7](#)

jsonStr, [7](#)

listExtract, [8](#)

namesToString, [8](#)

pasteCols, [9](#)
pastePaths, [10](#)

quoteText, [10](#)

sampleStr, [11](#)