Package 'splutil'

July 23, 2025

Title Utility Functions for Common Base-R Problems Relating to Lists

Version 2022.6.20

Description Utility functions that help with common base-R problems relating to lists. Lists in base-R are very flexible. This package provides functions to quickly and easily characterize types of lists. That is, to identify if all elements in a list are null, data.frames, lists, or fully named lists. Other functionality is provided for the handling of lists, such as the easy splitting of lists into equally sized groups, and the unnesting of data.frames within fully named lists.

Depends R (>= 3.3.0)

License MIT + file LICENSE

URL https://docs.sykdomspulsen.no/splutil/, https://github.com/sykdomspulsen-org/splutil

BugReports https://github.com/sykdomspulsen-org/splutil/issues

Encoding UTF-8

Imports data.table, magrittr, ggplot2

Suggests testthat, knitr, rmarkdown, rstudioapi, glue

RoxygenNote 7.1.2

VignetteBuilder knitr

NeedsCompilation no

Author Richard Aubrey White [aut, cre] (ORCID: <https://orcid.org/0000-0002-6747-1726>), Folkehelseinstituttet [cph]

Maintainer Richard Aubrey White <sykdomspulsen@fhi.no>

Repository CRAN

Date/Publication 2022-06-22 07:10:02 UTC

Contents

`easy_split` *Split a vector into a list of vectors*

Description

Easily split a list into a list of equally sized vectors.

Usage

```
easy_split(x, size_of_each_group = NULL, number_of_groups = NULL)
```

Arguments

`x`	The vector to be split
`size_of_each_group`	If you want to split ‘x‘ into a number of groups, each of ‘size_of_each_group‘ size
`number_of_groups`	How many equally sized groups do you want?

Details

You can either specify the length of the list (via ‘number_of_groups‘) or the length of the equally sized vectors within each list element (via ‘size_of_each_group‘). The last element of the list can be shorter than the other elements.

Value

A list containing equally sized vectors.

Examples

```
easy_split(letters[1:20], size_of_each_group = 3)
easy_split(letters[1:20], number_of_groups = 3)
```

is_all_list_elements_null_or_df

Are all elements in a list null or data.frames?

Description

Checks if A) 'x' is a list, B) All elements in 'x' are either null or data.frame.

Usage

```
is_all_list_elements_null_or_df(x)
```

Arguments

x	An object

Value

Boolean.

Examples

```
is_all_list_elements_null_or_df(data.frame())
is_all_list_elements_null_or_df(list(data.frame()))
is_all_list_elements_null_or_df(list(1, NULL))
is_all_list_elements_null_or_df(list(data.frame(), NULL))
is_all_list_elements_null_or_df(list("a"=1, 2))
```

is_all_list_elements_null_or_fully_named_list

Are all elements in a list null or fully named lists?

Description

Checks if A) 'x' is a list, B) All elements in 'x' are either null or fully named lists.

Usage

```
is_all_list_elements_null_or_fully_named_list(x)
```

Arguments

x	An object

Details

Fully named lists are lists with each element having a name.

Value

Boolean.

Examples

```
is_all_list_elements_null_or_fully_named_list(data.frame())
is_all_list_elements_null_or_fully_named_list(list(data.frame()))
is_all_list_elements_null_or_fully_named_list(list(1, NULL))
is_all_list_elements_null_or_fully_named_list(list(list(), NULL))
is_all_list_elements_null_or_fully_named_list(list(list("a" = 1), NULL))
is_all_list_elements_null_or_fully_named_list(list("a"=1, 2))
```

---

is_all_list_elements_null_or_list

Are all elements in a list null or lists?

Description

Checks if A) ‘x‘ is a list, B) All elements in ‘x‘ are either null or list.

Usage

```
is_all_list_elements_null_or_list(x)
```

Arguments

x	An object

Value

Boolean.

Examples

```
is_all_list_elements_null_or_list(data.frame())
is_all_list_elements_null_or_list(list(data.frame()))
is_all_list_elements_null_or_list(list(1, NULL))
is_all_list_elements_null_or_list(list(list(), NULL))
is_all_list_elements_null_or_list(list("a"=1, 2))
```

is_fully_named_list *Is this a fully named list?*

Description

Checks if ‘x‘ is a list with each element named.

Usage

```
is_fully_named_list(x)
```

Arguments

x	An object

Value

Boolean.

Examples

```
is_fully_named_list(list())
is_fully_named_list(list(1))
is_fully_named_list(list("a"=1))
is_fully_named_list(list("a"=1, 2))
```

unnest_dfs_within_list_of_fully_named_lists *Unnest data.frames within fully named list*

Description

Consider the situation where a function returns a list containing two data.frames. If this function is called repeatedly and the return values are stored in a list, we will have a list of fully named lists (each of which contains a data.frame). Typically, we want to extract the two data.frames from this nested list structure (and rbindlist them).

Usage

```
unnest_dfs_within_list_of_fully_named_lists(
  x,
  returned_name_when_dfs_are_not_nested = "data",
  ...
)
```

Arguments

x	A list of fully named lists (which then contain data.frames)
returned_name_when_dfs_are_not_nested	When x is a single list of data.frames, what name should be returned?
...	parameters passed to data.table::rbindlist

Value

Fully named list, each element containing a data.table.

Examples

```
x <- list(
  list(
    "a" = data.frame("v1"=1),
    "b" = data.frame("v2"=3)
  ),
  list(
    "a" = data.frame("v1"=10),
    "b" = data.frame("v2"=30),
    "d" = data.frame("v3"=50)
  ),
  list(
    "a" = NULL
  ),
  NULL
)
print(x)
splutil::unnest_dfs_within_list_of_fully_named_lists(x)

x <- list(
  data.frame("v1"=1),
  data.frame("v3"=50)
)
print(x)
splutil::unnest_dfs_within_list_of_fully_named_lists(
  x,
  returned_name_when_dfs_are_not_nested = "NAME",
  fill = TRUE
)
```

Index