Package ‘dapr’

May 6, 2019

Title 'purrr'-Like Apply Functions Over Input Elements

Version 0.0.3

Description An easy-to-use, dependency-free set of functions for iterating over elements of various input objects. Functions are wrappers around base apply()/lapply()/vapply() functions but designed to have similar functionality to the mapping functions in the 'purrr' package <https://purrr.tidyverse.org/>. Specifically, function names more explicitly communicate the expected class of the output and functions also allow for the convenient shortcut of '~ .x' instead of the more verbose 'function(.x) .x'.

License GPL-3

Encoding UTF-8

Language en

LazyData true

RoxygenNote 6.1.1

URL `https://github.com/mkearney/dapr`

BugReports `https://github.com/mkearney/dapr/issues`

Suggests testthat, covr

NeedsCompilation no

Author Michael W. Kearney [aut, cre] (<https://orcid.org/0000-0002-0730-4694>)

Maintainer Michael W. Kearney <`kearneymw@missouri.edu`>

Repository CRAN

Date/Publication 2019-05-06 17:10:03 UTC

R topics documented:

dap *dap: Data frame apply functions*

Description

Functions that apply expressions to input data objects and return data frames.

dapc: Apply function to columns of a data frame.

dapr: Apply function to rows of a data frame.

dapc_if: Apply function to certain columns of a data frame.

dapr_if: Apply function to certain rows of a data frame.

Usage

```
dapc(.data, .f, ...)

dapr(.data, .f, ...)

dapc_if(.data, .predicate, .f, ...)

dapr_if(.data, .predicate, .f, ...)
```

Arguments

`.data`	Data frame input.
`.f`	Function to apply to element (columns or rows). This can be written as a single function name e.g., `mean`, a formula-like function call where '.x' is assumed to be the iterated over element of input data e.g., `~ mean(.x)`, or an in-line function definition e.g., `function(x) mean(x)`.
`...`	Other values passed to function call.
`.predicate`	Logical vector or expression evaluating to a logical vector. If not a logical vector, this can be written as a single function name e.g., `is.numeric`, a formula-like function call where '.x' is assumed to be the iterated over element of input data e.g., `~ is.numeric(.x)`, or an in-line function definition e.g., `function(x) is.numeric(x)`. Regardless, if a logical vector is not provided, this expression must return a logical vector of the same length as the input .data object. The resulting logical vector is used to determine which elements (rows or columns) to iterate over with the .f function/expression.

Value

A data frame

See Also

lap vap

`ilap` *Iterator list apply*

Description

ilap: Iterate over sequence length of input and return list(s)

Usage

```
ilap(.data, .f, ...)
```

Arguments

`.data`	Object representing the sequence length, used as ".i" in the .f expression. If matrix or data frame, sequence length will be determined by the number of columns. If integer, then it will be passed directly onto function call, otherwise the object will be converted into a sequence from 1 to the length of the object.
`.f`	Function to apply to each element (as an integer position) of input object. This can be written as a single function name e.g., `mean`, a formula-like function call where '.i' is assumed to be the integer position of input data object e.g., `~ mean(mtcars[[.i]])`, or an in-line function definition e.g., `function(x) mean(mtcars[[x]])`.
`...`	Other values passed to function call.

Value

A list

See Also

Other lap: `lap`

Examples

```

## return string list
ilap(1:10, ~ paste0(letters[.i], rev(LETTERS)[.i]))

## return list of columns
ilap(mtcars, ~ c(row.names(mtcars)[.i], mtcars$wt[.i]))
```

lap — *lap: List apply functions*

Description

Function(s) that apply expressions to input data objects and return lists.

lap: Iterate over input and return list(s)

Usage

```
lap(.data, .f, ...)
```

Arguments

`.data`	Input object–numeric, character, list, data frame, etc.–over which elements will be iterated. If matrix or data frame, each column will be treated as the elements which are to be iterated over.
`.f`	Function to apply to each element of input object. This can be written as a single function name e.g., `mean`, a formula-like function call where '.x' is assumed to be the iterated over element of input data e.g., `~ mean(.x)`, or an in-line function definition e.g., `function(x) mean(x)`.
`...`	Other values passed to function call.

Value

A list

See Also

`dap` `vap`

Other lap: `ilap`

Examples

```
## return string list
lap(letters, ~ paste0(.x, "."))

## return list of columns
lap(mtcars[1:5, ], as.character)
```

vap *vap: Vector apply functions*

Description

Functions that apply expressions to input data objects and return atomic vectors e.g., numeric (double), character, logical.

vap_dbl: Iterate over input and return double(s)

vap_chr: Iterate over input and return character(s)

vap_lgl: Iterate over input and return logical(s)

vap_int: Iterate over input and return integer(s)

Usage

```
vap_dbl(.data, .f, ...)

vap_chr(.data, .f, ...)

vap_lgl(.data, .f, ...)

vap_int(.data, .f, ...)
```

Arguments

`.data`	Input object–numeric, character, list, data frame, etc.–over which elements will be iterated. If matrix or data frame, each column will be treated as the elements which are to be iterated over.
`.f`	Function to apply to each element of input object. This can be written as a single function name e.g., `mean`, a formula-like function call where '.x' is assumed to be the iterated over element of input data e.g., `~ mean(.x)`, or an in-line function definition e.g., `function(x) mean(x)`.
`...`	Other values passed to function call.

Value

A double vector

A character vector

A logical vector

A integer vector

See Also

dap lap

Examples

```
## character
vap_chr(letters, ~ paste0(.x, "."))

## double
vap_dbl(rnorm(4), round, 3)

## logical
vap_lgl(letters, ~ .x %in% c("a", "e", "i", "o", "u"))

## integer
vap_int(rnorm(5), ~ as.integer(.x))
```

Index