

Package ‘wildcard’

July 22, 2025

Title Templates for Data Frames

Version 1.1.0

Date 2017-09-16

Description Generate data frames from templates.

License GPL (>= 3)

Depends R (>= 3.0.0)

Imports magrittr, stringi

Suggests knitr, rmarkdown, testthat

URL <https://github.com/wlandau/wildcard>

BugReports <https://github.com/wlandau/wildcard/issues>

Encoding UTF-8

RoxygenNote 6.0.1

VignetteBuilder knitr

NeedsCompilation no

Author William Michael Landau [aut, cre]

Maintainer William Michael Landau <will.landau@gmail.com>

Repository CRAN

Date/Publication 2017-09-16 11:49:05 UTC

Contents

wildcard-package	2
expandrows	2
nofactors	3
wildcard	4

Index	6
--------------	----------

wildcard-package

*A templating mechanism for data frames***Description**

A templating mechanism for data frames

Author(s)

William Michael Landau <will.landau@gmail.com>

References

<https://github.com/wlandau/wildcard>

Examples

```
myths <- data.frame(
  myth = c('Bigfoot', 'UFO', 'Loch Ness Monster'),
  claim = c('various', 'day', 'day'),
  note = c('various', 'pictures', 'reported day'))
wildcard(myths, wildcard = 'day', values = c('today', 'yesterday'))
wildcard(myths, wildcard = 'day', values = c('today', 'yesterday'),
  expand = FALSE)
locations <- data.frame(
  myth = c('Bigfoot', 'UFO', 'Loch Ness Monster'),
  origin = 'where')
rules <- list(
  where = c('North America', 'various', 'Scotland'),
  UFO = c('spaceship', 'saucer'))
wildcard(locations, rules = rules, expand = c(FALSE, TRUE))
numbers <- data.frame(x = 4, y = 3, z = 4444, w = 4.434)
wildcard(numbers, wildcard = 4, values = 7)
df <- data.frame(
  ID = c('24601', 'Javert', 'Fantine'),
  fate = c('fulfillment', 'confusion', 'misfortune'))
expandrows(df, n = 2, type = 'each')
expandrows(df, n = 2, type = 'times')
```

expandrows

*Function expand***Description**

Expand the rows of a data frame Copied and modified from `remakeGenerator::expand()` under GPL>=3: <https://github.com/wlandau/remakeGenerator>

Usage

```
expandrows(df, n = 2, type = c("each", "times"))
```

Arguments

df	data frame
n	number of duplicates per row
type	character scalar. If 'each', rows will be duplicated in place. If 'times', the data frame itself will be repeated n times.

See Also

[wildcard](#)

Examples

```
df <- data.frame(  
  ID = c('24601', 'Javert', 'Fantine'),  
  fate = c('fulfillment', 'confusion', 'misfortune'))  
expandrows(df, n = 2, type = 'each')  
expandrows(df, n = 2, type = 'times')
```

nofactors

Function nofactors

Description

Turn all the factors of a data frame into characters.

Usage

```
nofactors(df)
```

Arguments

df	data frame
----	------------

See Also

[wildcard](#)

Examples

```
class(iris$Species)  
str(iris)  
out <- nofactors(iris)  
class(out$Species)  
str(out)
```

wildcard	<i>Function</i> wildcard
----------	--------------------------

Description

Main function of the package. Evaluate a wildcard to fill in or expand a data frame. Copied and modified from `remakeGenerator::evaluate()` under GPL-3: <https://github.com/wlandau/remakeGenerator>

Usage

```
wildcard(df, rules = NULL, wildcard = NULL, values = NULL,
  expand = TRUE, include = NULL, exclude = NULL)
```

Arguments

<code>df</code>	data frame
<code>rules</code>	list with names a wildcards and elements as vectors of values to substitute in place of the wildcards.
<code>wildcard</code>	character scalar, a wildcard found in a data frame
<code>values</code>	vector of values to substitute in place of a wildcard
<code>expand</code>	logical, whether to expand the rows of the data frame to substitute each value for each wildcard in turn. If FALSE, no new rows will be added to <code>df</code> when the values are substituted in place of wildcards. Can be a vector of length <code>length(rules)</code> if using the <code>rules</code> argument.
<code>include</code>	character vector of columns of <code>df</code> to be included in the wildcard evaluation. The values will replace the wildcards in these columns but not in any of the other columns. All columns are included by default. You may use <code>include</code> or <code>exclude</code> (or neither), but not both.
<code>exclude</code>	character vector of columns of <code>df</code> to be EXCLUDED from the wildcard evaluation. The values will NOT replace the wildcards in any of these columns, but wildcard evaluation will occur in all the other columns. By default, no columns are excluded (all columns are used for wildcard evaluation). You may use <code>include</code> or <code>exclude</code> (or neither), but not both.

Examples

```
myths <- data.frame(
  myth = c('Bigfoot', 'UFO', 'Loch Ness Monster'),
  claim = c('various', 'day', 'day'),
  note = c('various', 'pictures', 'reported day'))
wildcard(myths, wildcard = 'day', values = c('today', 'yesterday'))
wildcard(myths, wildcard = 'day', values = c('today', 'yesterday'),
  expand = FALSE)
locations <- data.frame(
  myth = c('Bigfoot', 'UFO', 'Loch Ness Monster'),
```

```
origin = 'where')
rules <- list(
  where = c('North America', 'various', 'Scotland'),
  UFO = c('spaceship', 'saucer'))
wildcard(locations, rules = rules, expand = c(FALSE, TRUE))
numbers <- data.frame(x = 4, y = 3, z = 4444, w = 4.434)
wildcard(numbers, wildcard = 4, values = 7)
# Inclusion and exclusion
wildcard(myths, wildcard = "day", values = c("today", "yesterday"),
  include = "claim")
wildcard(myths, wildcard = "day", values = c("today", "yesterday"),
  exclude = c("claim", "note"))
# Wildcards should not also be replacement values.
# Otherwise, the output will be strange
# and will depend on the order of the wildcards.
## Not run:
df <- data.frame(x = "a", y = "b")
rules <- list(a = letters[1:3], b = LETTERS[1:3])
wildcard(df, rules = rules)

## End(Not run)
```

Index

expandrows, [2](#)

nofactors, [3](#)

wildcard, [3](#), [4](#)

wildcard-package, [2](#)