Package ‘grapes’

October 13, 2022

Title Make Binary Operators
Version 1.0.0
Date 2017-04-01
Description Turn arbitrary functions into binary operators.
License GPL (>= 3)
Depends R (>= 3.0.0)
Imports magrittr
Suggests knitr, rmarkdown, testthat
VignetteBuilder knitr

URL https://github.com/wlandau/grapes

BugReports https://github.com/wlandau/grapes/issues
Encoding UTF-8
RoxygenNote 5.0.1
NeedsCompilation no

Author William Michael Landau [aut, cre, cph]
Maintainer William Michael Landau <will.landau@gmail.com>
Repository CRAN
Date/Publication 2017-04-01 16:22:11 UTC

R topics documented:

bunch ................................................................. 2
functions ........................................................... 2
grow ................................................................. 3

Index 4
functions

bunch | Function bunch

Description
List the available operators in a package or environment. The environment defaults to your workspace.

Usage
bunch(from = parent.frame())

Arguments
from package (character scalar) or environment to look for operators

Value
character vector of all the available operators in an environment

Examples
grow(rbind, from = "base")
bunch() # has "%rbind%"
myfun = function(x, y) x + y
env = new.env()
grow(myfun, to = env)
bunch() # has "%rbind%"
bunch(from = env) # "%myfun%"

functions | Function functions

Description
List the available functions in a package or environment that are not already binary operators.

Usage
functions(from = parent.frame())

Arguments
from package (character scalar) or environment to look for functions

Value
character vector of the names of available functions in from
Examples

```r
functions()
myfun = function(x, y) x + y
functions()
functions("grapes")
functions("knitr")
```

---

**grow**

**Function grow**

Description

Turn a collection of functions into binary operators. For example, create the binary operator `%cbind%` from the function `cbind()` by calling `grow(cbind)`. Then, you can call `sleep %cbind% sleep %cbind% sleep` to add to the columns of a data frame while avoiding cumbersome parentheses.

Usage

```r
grow(..., list = character(0), from = parent.frame(), to = parent.frame())
```

Arguments

- `...`: characters or symbols, names of functions to make into binary operators.
- `list`: character vector of names of functions to make into binary operators
- `from`: package (character scalar) or environment to look for functions to turn into binary operators.
- `to`: environment to store the new binary operators.

Examples

```r
grow(rbind, from = "base") # Use 'from' to specify a package or environment to search.
nrow(sleep) # 20
longer = sleep %rbind% sleep %rbind% sleep # No clumsy parentheses!
nrow(longer) # 60. Most of us would like to sleep longer.
```
Index

bunch, 2

cbind, 3

functions, 2

grow, 3