Package ‘refset’

October 14, 2022

Type Package
Title Subsets with Reference Semantics
Version 0.1.1
Date 2014-11-20
Author ``David Hugh-Jones'' <davidhughjones@gmail.com>
Maintainer ``David Hugh-Jones'' <davidhughjones@gmail.com>
URL http://github.com/hughjonesd/refset
Description Provides subsets with reference semantics, i.e. subsets
which automatically reflect changes in the original object, and which
optionally update the original object when they are changed.
License GPL-2
NeedsCompilation no
VignetteBuilder knitr
Suggests knitr, rmarkdown, testthat
RoxygenNote 6.0.1
Repository CRAN
Date/Publication 2017-12-10 15:10:07 UTC

R topics documented:

refset-package .................................................. 2
contents .......................................................... 2
is.parcel .......................................................... 3
refset .............................................................. 4
unwrap_as ........................................................ 6
wrap ................................................................. 6
wrapset ............................................................ 7

Index 9
refset-package  Subsets with Reference Semantics

Description
Subsets with Reference Semantics

Details
Refset provides subsets with reference semantics, i.e. subsets which automatically reflect changes in the original object, and which also update the original object when they are changed.
For more info see vignette('refset').

Author(s)
David Hugh-Jones <davidhughjones@gmail.com>

contents  Returns or changes parcel contents

Description
contents returns the value of the parcel contents by evaluating the expression in the parcel. contents<- attempts to assign to the expression, which will only work if the expression is appropriate, e.g. a refset.

Usage
contents(parcel)

contents(parcel) <- value

Arguments
parcel  an object of class 'parcel'
value  a value to assign

Value
The result of evaluating the expression stored in the parcel. For contents<-, the parcel itself. contents<- will only work if the expression wrapped in the parcel can accept assignments.

See Also
Other wrapping functions: is.parcel, unwrap_as, wrapset, wrap
is.parcel

Examples

pcl <- wrap(x^2)
x <- 2
contents(pcl)
x <- 3
contents(pcl)
## Not run:
contents(pcl) <- 4 # fails

## End(Not run)
p2 <- wrap(names(x))
contents(p2) <- "named"
x


is.parcel  Checks whether an object is a parcel

Description

Checks whether an object is a parcel

Usage

is.parcel(x)

Arguments

x  an object to examine

Value

TRUE or FALSE.

See Also

Other wrapping functions: contents, unwrap_as, wrapset, wrap
refset

Create a reference to a subset of an object

Description

Create a refset - a reference to a subset of an object. When the object changes, the contents of the refset change, and when the refset is changed, the object is changed too.

Usage

refset(x, data, ..., drop = TRUE, dyn.idx = TRUE, read.only = FALSE, eval.env = parent.frame(), assign.env = parent.frame())

x %r% data

Arguments

x
name of the refset to create, as a bare name or character string

data
the object to refer to

...
indices to subset with

drop
passed to Extract

dyn.idx
update indices dynamically

read.only
create a read-only refset which throws an error if assigned to

eval.env
environment in which data and indices will be evaluated

assign.env
environment in which the variable named by x will be created

Details

There are two ways to call refset. The two-argument form, e.g. refset(myref, mydata[rows,"column"]), creates a reference to the subset of mydata passed in the second argument.

The three-or-more argument form acts like the subset function: the indices in ... are applied to data. If data is a data.frame, then the indices are interpreted within it, so you can refer to columns directly: refset(myref, mydata, a>1 & b<a,). Bare column names must be quoted, however.

Empty arguments in ... are allowed and are treated as indexing the whole dimension, just as in Extract.

By default, the indices in subset are updated dynamically. For example, if you call refset(myref, mydata, x >= 3,) and then set mydata$x <- 3, the number of rows in myref will probably increase. To turn this behaviour off and make a reference to a “fixed” subset of your object, use dyn.idx=FALSE.

%r% is an infix version of the two-argument form.

Value

refset returns NULL, but the x argument will be assigned to in the calling environment (or in env, if it is specified). x will have an attribute ".refset.".
See Also

Refsets are implemented using makeActiveBinding.

Examples

dfr <- data.frame(a=1:4, b=1:4)
s <- dfr[1:2,]
refset(rs, dfr[1:2,])
dfr$a <- 4:1
s # 1:2
rs # 4:3

# same:
refset(rs, dfr, 1:2, )

# same:
rs %r% dfr[1:2,]

vec <- 1:10
refset(middle, vec[4:6])
middle <- 4:6 + 100
vec

# dynamic versus static indices:
dfr <- data.frame(a=rnorm(100), b=rnorm(100))
refset(ss, dfr, a>1,)
refset(ss.static, dfr, a>1,, dyn.idx=FALSE)
nrow(ss) == nrow(ss.static)
dfr$a <- dfr$a + 2 * dfr$b

precious.data <- rnorm(100)
refset(big, precious.data, precious.data>1, read.only=TRUE)
big
## Not run:
big <- big * 2 # throws an error

## End(Not run)

# Using refset with other functions:
# dynamically updated calculated column
dfr <- data.frame(a=rnorm(10), b=rnorm(10))
refset(rs, transform(dfr, x=a+2*b+rnorm(10)))
rs
rs # different

# Non-readonly refset with other functions. Works but gives a warning:
## Not run:
vec <- 1:5
refset(ssv, names(vec), read.only=FALSE)
unwrap_as <- LETTERS[1:5]
vec

## End(Not run)

unwrap_as

Unwrap contents of a parcel into a new variable

Description
unwrap_as creates a new variable which, when evaluated, calls contents to return the parcel contents.

Usage
unwrap_as(x, parcel, env = parent.frame())

Arguments
x name of the variable to bind to
parcel an object of class 'parcel'
env environment to assign the variable into

See Also
Other wrapping functions: contents, is.parcel, wrapset, wrap

Examples
vec <- 1:10
parcel <- wrapset(vec, vec > 3)
unwrap_as(y, parcel)

wrap

Wrap an expression and its environment into a parcel.

Description
Refsets (and other active bindings) cannot be passed as function arguments, since doing so makes a copy. wrap allows you to pass arbitrary expressions between functions and records where they are ultimately evaluated.

Usage
wrap(expr, env = parent.frame())
Arguments

expr an R expression
env environment in which expr is to be evaluated

Value

An object of class 'parcel', with components expr and env.

See Also

Other wrapping functions: contents, is.parcel, unwrap.as, wrapset

Examples

dfr <- data.frame(a=1:4, b=1:4)
rs %r% dfr[1:2,]
parcel <- wrap(rs)
f <- function (parcel) contents(parcel) <- contents(parcel)*2
f(parcel)
contents(parcel)
dfr

parcel <- wrap(x^2) # non-refset use
x <- 3
f <- function(parcel) {x <- 10; contents(parcel)}
f(parcel)

wrapset

Convenience function to create a parcel containing a refset.

Description

wrapset calls refset on its arguments and returns the resulting active binding in a parcel object for passing around.

Usage

wrapset(data, ..., env = parent.frame())

Arguments

data, ... passed to refset
env passed to refset as argument eval.env

Value

A parcel object.
See Also

Other wrapping functions: `contents`, `is.parcel`, `unwrap_as`, `wrap`

Examples

```r
dfr <- data.frame(a=1:5, b=1:5)
parcel <- wrapset(dfr, a<3, , drop=FALSE)
contents(parcel)
```
Index

* package
  refset-package, 2
%r% (refset), 4
‘%r%‘ (refset), 4
contents, 2, 3, 6–8
contents<- (contents), 2
Extract, 4
is.parcel, 2, 3, 6–8
refset, 4, 7
refset-package, 2
subset, 4
unwrap_as, 2, 3, 6, 7, 8
wrap, 2, 3, 6, 6, 8
wrapset, 2, 3, 6, 7, 7