Package 'tribe'

August 2, 2018

Type Package

Title Play with the Tribe of Attributes

Version 0.1.7

Date 2018-08-02

Description Functions to make manipulation of object attributes easier.
It also contains a few functions that extend the 'dplyr' package for data manipulation, and it provides new pipe operators, including the pipe '%@>%' similar to the 'magrittr' '%>%', but with the additional functionality to enable attributes propagation.

License MIT + file LICENSE

LazyData TRUE

Depends R (>= 3.1.3)

Imports bazar, dplyr, lazyeval, magrittr, rlist, rstudioapi

VignetteBuilder knitr

Suggests knitr, ggplot2, lplyr, observer, testthat

URL https://github.com/paulponcet/tribe

BugReports https://github.com/paulponcet/tribe/issues

RoxygenNote 6.0.1

NeedsCompilation no

Author Paul Poncet [aut, cre],
Stefan Milton Bache [aut] (for functions copied or modified from the 'magrittr' package),
Hadley Wickham [aut] (for functions copied or modified from the 'magrittr' package)

Maintainer Paul Poncet <paulponcet@yahoo.fr>

Repository CRAN

Date/Publication 2018-08-02 11:10:02 UTC

R topics documented:

`at` *Infix attribute accessor*

Description

Infix attribute accessor

Usage

```
x %@% name
```

Arguments

`x`	Object
`name`	Attribute name

Author(s)

Borrowed from Hadley Wickham's **purrr** package.

Examples

```
factor(1:3) %@% "levels"
factor(1:3) %@% levels

mtcars %@% "class"
mtcars %@% class
```

`at_mutate` *Manipulate attributes in a dplyr fashion*

Description

The function `at_mutate` adds or changes attributes to `obj`.

The function `at_select` selects attributes of `obj`, and removes the others.

The function `at_rename` renames attributes of `obj`.

The function `at_slice` chooses a specific attribute and returns it.

Usage

```
at_mutate(obj, ...)

at_mutate_(obj, ..., .dots)

at_select(obj, ...)

at_select_(obj, ..., .dots)

at_rename(obj, ...)

at_rename_(obj, ..., .dots)

at_slice(obj, at)

at_slice_(obj, at)
```

Arguments

`obj`	An object.
`...`	Comma separated list of unquoted expressions.
`.dots`	Used to work around non-standard evaluation.
`at`	Attribute to be obtained.

Value

`at_slice` returns the attribute chosen. The other functions return `obj` with possibly modified attributes.

See Also

`structure`, `attributes`

Examples

```
library(dplyr)
df <- data.frame(x = sample(10, 5, rep = TRUE),
                 y = sample(10, 5, rep = TRUE)) %>%
  at_mutate(example="yes",
            package="dplyr")
tribe(df)

at_slice(df, names)
at_slice_(df, "class")
at_slice_(df, ~ package)

df <- df %>%
  at_mutate_(package = ~ NULL,
             example = ~ "no")
tribe(df)

df <- df %>%
  at_mutate_(.dots = list(x =~ 2, y =~ c(3,4)))
tribe(df)
```

`make_pipe` *Create a pipe operator.*

Description

This function is used to create magrittr like pipe operators.

Usage

```
make_pipe(propagate, keep_also = NULL, try = FALSE)

lhs %@>% rhs

lhs %<@>% rhs

lhs %try>% rhs
```

Arguments

`propagate`	character. See the eponymous argument in `shield`.
`keep_also`	character. See the eponymous argument in `shield`.
`try`	logical. If `TRUE` and the pipe `x > f` generates an error, then the pipe `x try> f` returns `x` unchanged silently.
`lhs`	Left-hand side of the pipe.
`rhs`	Right-hand side of the pipe.

Author(s)

Stefan Milton Bache and Hadley Wickham for the original `pipe` function in package **magrittr**; Paul Poncet for the modifications introduced.

See Also

`shield` in this package.

Examples

```
library(dplyr)
df <- data.frame(x = sample(10, 5, rep = TRUE),
                 y = sample(10, 5, rep = TRUE)) %>%
  at_mutate(example="yes",
            package="dplyr",
            class = c("my_tbl", "data.frame"))
tribe(df)

# Attributes just created are lost when the object
# passes through dplyr verbs
tribe(df %>% mutate(z=3))

# With the pipe '%@>%', most attributes are kept
tribe(df %@>% mutate(z=3))

# One can create a new pipe to adjust attributes propagation settings
"%newpipe>%" <- make_pipe(propagate="none", keep_also = "example")
tribe(df %newpipe>% mutate(z=3))
```

`shield` *Attributes protection*

Description

The function `shield` is made to facilitate the propagation of attributes of an object `obj` through R operations.

Usage

```
shield(obj, at, propagate = "some", keep_also = NULL)
```

Arguments

`obj`	An object.
`at`	A named list, the attributes to be possibly added to `obj`.

propagate	character. The method to be applied, one of "all", "most", "some", "none", "many". If propagate="some" (the default), the attributes of obj are kept unchanged (up to the value of keep_also). If propagate="all" (not advised), the attributes of the returned object are exactly at (up to the value of keep_also). If propagate="none" (not advised either), the attributes of the returned object are NULL (up to the value of keep_also). If propagate="most", new attributes taken from at will be added to obj; however, attributes found in at that have the same name as attributes of obj are not considered.
keep_also	character. A vector of named attributes to be added to the final result.

Value

The object obj with possibly different attributes.

Examples

```
library(dplyr)
df <- data.frame(x = sample(10, 5, rep = TRUE),
                 y = sample(10, 5, rep = TRUE)) %>%
  at_mutate(example="yes",
            package="dplyr",
            class = c("my_tbl", "data.frame"))
tribe(df)

# Attributes are lost when the object passes through dplyr verbs
df2 <- df %>%
  mutate(z = 3)
tribe(df2)

# Most attributes are kept
df3 <- shield(df2, tribe(df), propagate = "most")
tribe(df3)

# To keep the class, use 'keep_also'
df4 <- shield(df2, tribe(df), propagate = "most", keep_also = "class")
tribe(df4)
```

---

stick_to *Work on a specific attribute within a pipeline*

Description

The functions stick_to and unstick enable to select an attribute within a pipe and work on it.

It must be combined with the %@>% pipe to work properly, see the example below.

Usage

```
stick_to(obj, at)

stick_to_(obj, at)

unstick(x)
```

Arguments

obj	An object with an at attribute.
at	The name of the attribute to be considered.
x	An object to be unsticked. Must have ".obj_stick" and ".at_stick" attributes.

Value

stick_to basically inverses the roles of .data and at, meaning that .data becomes an attribute of the selected attribute. unstick makes the inverse operation.

Examples

```
library(dplyr)
library(observer)

df <- ggplot2::diamonds %>%
  mutate(depth2 = 100*2*z/(x+y)) %>%
  observe_if(abs(depth-depth2) < 1)

observations(df)

df %>%
  stick_to(observations) %@>%
  mutate(Id = 2) %@>%
  select(Id, Status) %>%
  unstick()

observations(df)
```

---

tribe *Object attribute list*

Description

The function tribe is identical to attributes, expect that it always returns a named list (thus, when attributes will return NULL, tribe will return an empty named list).

Usage

```
tribe(obj, keep_obj = FALSE)

tribe(obj) <- value

untribe(x)
```

Arguments

obj	An object.
keep_obj	logical. If TRUE, obj is passed as an attribute to the result (useful in combination of untribe).
value	An appropriate named list of attributes, or NULL.
x	A list (of attributes) to be untribed.

Value

A named list, the attributes of obj.

See Also

attributes, attributes<-, mostattributes<-.

Examples

```
library(lplyr)
A <- c(x = 1, y = 2, z = 3) %>%
  at_mutate(package = "trib?")
A %>%
  tribe(keep_obj = TRUE) %@>%
  mutate(package = "tribe") %>%
  untribe()
```

Index