Package 'aka'

June 2, 2025

Title Define Aliases for R Expressions

Version 0.2.0

Description Create aliases for other R names or arbitrarily complex R expressions. Accessing the alias acts as-if the aliased expression were invoked instead, and continuously reflects the current value of that expression: updates to the original expression will be reflected in the alias; and updates to the alias will automatically be reflected in the original expression.

URL https://klmr.me/aka/, https://github.com/klmr/aka

BugReports https://github.com/klmr/aka/issues

Suggests testthat

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

Author Konrad Rudolph [cre, aut] (ORCID:
<https://orcid.org/0000-0002-9866-7051>)

Maintainer Konrad Rudolph <konrad.rudolph@gmail.com>

Depends R (>= 4.1.0)

Repository CRAN

Date/Publication 2025-06-02 15:50:02 UTC

Contents

`alias` *Create an alias for an expression*

Description

`alias(name = expr)` creates an alias for `expr` named `name`. Subsequently, `name` can (mostly) be used interchangeably with `expr`.

`name := expr` is the same as `alias(name = expr)`.

Usage

```
alias(name = expr, expr_env = parent.frame(), alias_env = parent.frame())

name := expr
```

Arguments

`expr_env`	the environment in which to evaluate the expression
`alias_env`	the environment in which to create the alias
`name`	the alias name
`expr`	an arbitrary R expression to be aliased by `name`; can contain interpolated expressions; see *Details*

Details

After executing `alias(name = expr)`, `name` can be used to refer to the value of `expr`. This is especially useful when `expr` is a complex expression that is used multiple times in the code. Unlike with regular assignment, `expr` will be reevaluated every time `name` is evaluated. This means that the value of `name` always stays up to date, similar to a "reactive" expression. On the flip side, it also means that accessing `name` can be very slow if evaluating `expr` is time-consuming.

`expr` can contain interpolated expressions using the `bquote()` syntax (including splicing). These will be substituted at the time of defining the alias. See *Examples*.

The parameters `expr_env` and `alias_env` are used to control the environments in which the expression is evaluated and the alias is created, respectively. Note that specifying the correct `expr_env` is particularly important when *assigning* to an alias: an expression can be evaluated inside a parent environment without having to specify `expr_env`; however, during assignment this would cause the assignee object to be copied into the calling environment. See *Examples* for a concrete example of this.

Value

`alias()` is called for its side-effect and does not return a value.

Examples

```
x = 'hello'
alias(ax = x)  # same as: ax := x
ax    # prints 'hello'

x = 'world'
ax    # prints 'world'

ax = 'goodbye'
x     # prints 'goodbye'

# Aliases can be created for complex expressions:
mercedes := mtcars[grepl('^Merc ', rownames(mtcars)), ]
mercedes

mercedes$vs = 0  # set all Mercedes engine types to V-shaped
mtcars

# Aliases can contain interpolated expressions:
n = 1
m = 2
s := .(n) + m
s  # prints 3

n = 10
m = 10
s  # prints 11

alias_expr('s')  # prints `1 + m`

# Be careful when assigning to an alias to an object in a parent environment:

e = attach(new.env())
e$y = 'hello'

ay := y

# Works: `y` is found in the parent environment
ay  # prints 'hello'

# But the following creates a *new variable* `y` in the current environment:
ay = 'world'
e$y   # prints 'hello', still!
y     # prints 'world'

# To prevent this, use the `expr_env` argument:
# alias(ay = y, expr_env = e)
```

`getters` *Query alias internals*

Description

`alias_expr(alias)` returns the expression that was used to define an alias.

`alias_env(alias)` returns the environment in which the aliased expression is evaluated.

Usage

```
alias_expr(alias, envir = parent.frame())

alias_env(alias, envir = parent.frame())
```

Arguments

`alias`	the name of an alias, as a string
`envir`	the environment in which to look up the alias name (default: calling environment)

Value

`alias_expr(alias)` returns an unevaluated R expression (a name or a call).

`alias_env(alias)` returns an environment.

Index