Package 'pointr'

July 23, 2025

Type Package

Title Working Comfortably with Pointers and Shortcuts to R Objects

Version 0.2.0

Maintainer Joachim Zuckarelli <joachim@zuckarelli.de>

Description R has no built-in pointer functionality. The 'pointr' package fills this gap and lets you create pointers to R objects, including subsets of dataframes. This makes your R code more readable and maintainable.

License GPL-3

BugReports https://github.com/jsugarelli/pointr/issues

URL https://github.com/jsugarelli/pointr/

Encoding UTF-8

LazyData true

Imports stringr

RoxygenNote 7.1.1

NeedsCompilation no

Author Joachim Zuckarelli [aut, cre] (ORCID: <https://orcid.org/0000-0002-9280-3016>)

Repository CRAN

Date/Publication 2021-01-06 04:50:02 UTC

Contents

pointr *Package 'pointr'*

Description

The **pointr** package allows to work with pointers to R objects/selection in order to make the R code more readable and maintainable. The main function of the package are: `ptr()` to create a pointer, `rm.ptr()` to remove a pointer, and `where.ptr()` to check the target object of a pointer.

ptr *Working with pointers*

Description

Create, remove and analyze pointers in R. Pointers can point to any R object, including selections/subsets.

Usage

```
ptr(symbol1, symbol2)

symbol1 %=% symbol2

rm.ptr(symbol1, keep = FALSE)

where.ptr(symbol1)
```

Arguments

`symbol1`	The name of the pointer, as a one-element character vector.
`symbol2`	The object/selection the pointer will point to, as a one-element character vector.
`keep`	A logical value relevant when removing a pointer with `rm.ptr`. If `TRUE`, the pointer variable will be kept and filled with a copy of the object the pointers points to; if `FALSE`, the pointer variable `symbol1` will be removed completely. Default is `FALSE`.

Details

The `ptr()` function and the `%=%` operator will create a pointer to an R object, like a vector, list, dataframe or even a subset/selection from a dataframe. `where.ptr()` shows where a pointer actually points to. Existing pointers can be removed usig the `rm.ptr()` function. Pointers created with **pointr** use active bindings that call a hidden access function everytime the pointer is accessed. This hidden access function is named `.pointer()` (where `pointer` is the name of the pointer variable) and is created in the environment from which `ptr()` is called. It is not necessary to call this hidden access function as a pointer user. The hidden access function is removed when `rm.ptr()` is called.

Value

`ptr()`, `%=%` and `rm.ptr()` have no return value. `ptr()` and `%=%` create the pointer variable (argument `symbol1`) in the environment from which it is called. `where.ptr` returns the object/selection a pointer points to as a character vector.

Contributions

Thanks to Chad Hammerquist for contributing the `pointr` operator `%=%`.

Examples

```
library(pointr)

# Pointer to simple variable

myvar <- 3
ptr("mypointer", "myvar")
mypointer

myvar <- 5
mypointer

mypointer <- 7
myvar


# Alternative: Use the pointr operator %=%

myvar <- 3
mypointr %=% myvar
myvar


# Pointer to subset from dataframe

df <- data.frame(list(var1 = c(1,2,3), var2 = c("a", "b", "c")), stringsAsFactors = FALSE)
df

i <- 2
ptr("sel", "df$var2[i]")

sel <- "hello"
df$var2[i]

df$var2[i] <- "world"
sel

where.ptr("sel")
```

Index