Package 'ehelp'

December 13, 2019

Title Enhanced Help to Enable ``Docstring"-Comments in Users Functions

Version 1.1.1

Author Marcelo Ponce [aut, cre]

Maintainer Marcelo Ponce <mponce@scinet.utoronto.ca>

Description By overloading the R help() function, this package allows users to use ``docstring" style comments within their own defined functions.

URL https://github.com/mponce0/eHelp

BugReports https://github.com/mponce0/eHelp/issues

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 6.1.99.9001

Suggests testthat (>= 2.1.0), knitr, rmarkdown, crayon

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2019-12-13 15:30:02 UTC

R topics documented:

help *Wrapper Help Function*

Description

This function is a wrapper around the R's system help() function. It allows the user to include docstring styles documentation and displayed it as help or information to the users using the help() command.

Usage

```
help(
  topic,
  package = NULL,
  lib.loc = NULL,
  verbose = getOption("verbose"),
  try.all.packages = getOption("help.try.all.packages"),
  help_type = getOption("help_type")
)
```

Arguments

topic	topic/or/function name to search for
package	package where to search
lib.loc	location of R libraries
verbose	for diplaying the filename
try.all.packages	attempt to go trough all installed packages
help_type	format of the displayed help (text,html, or pdf)

Details

Parameters are the same as in utils::help, see help(help,package='utils') for further details.

Examples

```
compute3Dveloc <- function(x,y,z,t){
#' @fnName compute3Dveloc
#' this function computes the velocity of an object in a 3D space
#' @param x  vector of positions in the x-axis
#' @param y  vector of positions in the y-axis
#' @param z  vector of positions in the z-axis
#' @param t  time vector corresponding to the position vector

   # number of elements in vectors
   n <- length(t)
   # compute delta_t
```

```
   delta_t <- t[2:n]-t[1:n-1]
   # compute delta_x
   delta_x <- x[2:n]-x[1:n-1]
   # compute delta_y
   delta_y <- y[2:n]-y[1:n-1]
   # compute delta_z
   delta_z <- z[2:n]-z[1:n-1]
   # do actual computation of velocity...
   veloc3D <- list(delta_x/delta_t, delta_y/delta_t, delta_z/delta_t)
   # return value
   return(veloc3D)
}

help(compute3Dveloc)
```

Index