Package 'ArgumentCheck'

August 29, 2016

Title Improved Communication to Users with Respect to Problems in Function Arguments

Version 0.10.2

Author Benjamin Nutter

Maintainer Benjamin Nutter <benjamin.nutter@gmail.com>

Description The typical process of checking arguments in functions is iterative. In this process, an error may be returned and the user may fix it only to receive another error on a different argument. 'ArgumentCheck' facilitates a more helpful way to perform argument checks allowing the programmer to run all of the checks and then return all of the errors and warnings in a single message.

Depends R (>= 3.0.0)

Suggests knitr, testthat

VignetteBuilder knitr

License GPL-3

LazyData true

URL https://github.com/nutterb/ArgumentCheck

BugReports https://github.com/nutterb/ArgumentCheck/issues/

RoxygenNote 5.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2016-04-01 22:47:22

R topics documented:

addError *Argument Checking*

Description

Checking function arguments can be done in a manner that allows all of the problems in the arguments to be noted and returned to the user in a single call. This avoids the potentially iterative process of finding problems in one argument, fixing the error, and going on to find problems in subsequent checks. The ArgCheck object can store messages for all of the problems and return these messages all at once, allowing the user the opportunity to fix all of the arguments before proceeding.

Usage

```
addError(msg, argcheck)

addMessage(msg, argcheck)

addWarning(msg, argcheck)

finishArgCheck(argcheck)

newArgCheck()
```

Arguments

msg	A Character string giving the message to return with an error or warning
argcheck	An object with class ArgCheck, usually created by newArgCheck

Details

newArgCheck initializes an ArgCheck object. This object stores the number of warnings, the number of errors, and the corresponding messages for the warnings and errors.

addError and addWarning are used to add messages to the ArgCheck objects.

finishArgCheck looks at the ArgCheck object. If it finds any errors or warnings, those are printed for the user to review. When errors are found, the function is terminated.

Author(s)

Benjamin Nutter

Examples

```
## Not run:
#* This example is taken from the discussion of argument checking at
#* http://www.r-bloggers.com/programming-with-r---checking-function-arguments/
cylinder.volume <- function(height, radius){
  Check <- newArgCheck()
```

```
  if (missing(height)){
    addError("A value for 'height' was not provided",
             Check)
  } else{
    if (height < 0)
      addError("'height' must be a non-negative number",
               Check)
  }

  if (missing(height)){
    addError("A value for 'radius' was not provided",
             Check)
  } else {
    if (radius < 0)
      addError("'radius' must be a non-negative number",
               Check)
  }

  if (!missing(height) & !missing(radius)){
    if (height < radius)
      addWarning("When 'height' < 'radius', you have a short, awkward looking cylinder",
                 Check)
  }

  finishArgCheck(Check)

  pi * radius^2 * height
}

cylinder.volume()
cylinder.volume(height = -3)
cylinder.volume(height = 3, radius = -2)
cylinder.volume(height = 3, radius=2)
cylinder.volume(height = -8, radius = 4)

## End(Not run)
```

ArgumentCheck *Improved Argument Check Communication*

Description

The typical process of checking arguments in functions is iterative. In this process, an error may be returned and the user may fix it only to receive another error on a different argument. 'ArgumentCheck 'facilitates a more helpful way to perform argument checks allowing the programmer to run all of the checks and then return all of the errors and warnings in a single message.

Source

The concepts of this package are heavily influenced by

Karafa, MT, "Building Better Macros: Basic Parameter Checking for Avoiding "ID10T" Errors. SAS Global Forum, 2011 http://support.sas.com/resources/papers/proceedings11/096-2011.pdf

Examples

```
## Not run:
#* This example is taken from the discussion of argument checking at
#* http://www.r-bloggers.com/programming-with-r---checking-function-arguments/
cylinder.volume <- function(height, radius){
  ArgCheck <- newArgCheck()
  ArgumentCheck::addError(missing(height),
           "A value for 'height' was not provided",
           ArgCheck)
  ArgumentCheck::addError(ifelse(!missing(height), height < 0, FALSE),
           "'height' must be a non-negative number",
           ArgCheck)
  ArgumentCheck::addError(missing(radius),
           "A value for 'radius' was not provided",
           ArgCheck)
  ArgumentCheck::addError(ifelse(!missing(radius), radius < 0, FALSE),
           "'radius' must be a non-negative number",
           ArgCheck)

  ArgumentCheck::addWarning(ifelse(!missing(height) & !missing(radius),
                    height < radius, FALSE),
            "When 'height' < 'radius', you have a short, awkward looking cylinder",
            ArgCheck)

ArgumentCheck::finishArgCheck(ArgCheck)

volume <- pi * radius^2 * height
volume
}

cylinder.volume()
cylinder.volume(height = -3)
cylinder.volume(height = 3, radius = -2)
cylinder.volume(height = 3, radius=2)
cylinder.volume(height = -8, radius = 4)

## End(Not run)
```

`match_arg` *Argument Verification Using Partial Matching*

Description

`match_arg` matches `arg` against a table of candidate values as specified by `choices`, where `NULL` means to take the first one. This is a modified version of the `base` package function `match.arg` that differs only in that it can interact with `ArgumentCheck` environments.

Usage

```
match_arg(arg, choices, several.ok = FALSE, argcheck)
```

Arguments

arg	a character vector (of length one unless several.ok is TRUE) or NULL.
choices	a character vector of candidate values
several.ok	logical specifying if arg should be allowed to have more than one element.
argcheck	An ArgumentCheck environment as returned by newArgCheck

Details

In the one-argument form match_arg(arg), the choices are obtained from a default setting for the formal argument arg of the function from which match.arg was called. (Since default argument matching will set arg to choices, this is allowed as an exception to the 'length one unless several.ok is TRUE' rule, and returns the first element.)

Matching is done using pmatch, so arg may be abbreviated.

Value

The unabbreviated version of the exact or unique partial match if there is one; otherwise, an error is signalled if several.ok is false, as per default. When several.ok is true and more than one element of arg has a match, all unabbreviated versions of matches are returned.

Author(s)

R Core. This function is a near-verbatim copy of the match.arg in base R.

See Also

pmatch, match.fun, match.call

Examples

```
require(stats)
# Extends the example for 'switch'
center <- function(x, type = c("mean", "median", "trimmed")) {
  type <- match.arg(type)
    switch(type,
           mean = mean(x),
           median = median(x),
           trimmed = mean(x, trim = .1))
}
x <- rcauchy(10)
center(x, "t")       # Works
center(x, "med")     # Works
try(center(x, "m"))  # Error
stopifnot(identical(center(x),       center(x, "mean")),
          identical(center(x, NULL), center(x, "mean")) )
```

```
## Allowing more than one match:
match.arg(c("gauss", "rect", "ep"),
          c("gaussian", "epanechnikov", "rectangular", "triangular"),
          several.ok = TRUE)
```

Index