Package ‘devFunc’

October 13, 2022

Type Package
Title Clear and Condense Argument Check for User-Defined Functions
Version 0.1
Author Robin Van Oirbeek
Maintainer Robin Van Oirbeek <robin.vanoirbeek@gmail.com>
Description A concise check of the format of one or multiple input arguments (data type, length or value) is provided. Since multiple input arguments can be tested simultaneously, a lengthy list of checks at the beginning of your function can be avoided, hereby enhancing the readability and maintainability of your code.
License GPL-3
Encoding UTF-8
Depends R (>= 3.3.0)
Imports plyr (>= 1.8.4), stringr (>= 1.1.0)
LazyData true
RoxygenNote 6.0.1
NeedsCompilation no
Repository CRAN
Date/Publication 2018-01-24 18:30:38 UTC

R topics documented:

  checkCharVec ................................. 2
  checkIntVec ................................. 2
  checkLength ................................. 3
  checkLogicVec ............................... 4
  checkNumOrIntVec ............................ 5
  checkNumVec ................................. 6
  checkRanges ................................. 6
  checkValues ................................. 7

Index 9
checkCharVec  

Checking if all elements of a list are all character vectors

Description
Checking if all elements of a list are all character vectors

Usage
checkCharVec(listChar, namesListElements = NULL)

Arguments
listChar
A list of the vectors of which one wishes to check if their data type is character

namesListElements
Character vector containing the names of the variables of which the data type is checked. Optional parameter, with as default value NULL. This argument should be used when the variable of which the data type is checked is not an object that was provided as an argument to the function, or when the list elements of the first argument do not have a name attached to it.

Value
No value is returned if all vectors have the character data type. If not, an error message is thrown for each element of the list that does not pertain to the character data type.

Examples
arg1 <- 'something'
checkCharVec(list(arg1))

checkCharVec(list('somethingElse', TRUE))

arg2 <- 2
checkCharVec(list(arg2))
checkCharVec(list(arg2, TRUE, 5L))

checkIntVec  

Checking if all elements of a list are all integer vectors

Description
Checking if all elements of a list are all integer vectors

Usage
checkIntVec(listInt, namesListElements = NULL)
checkLength

Arguments

listInt A list of the vectors of which one wishes to check if their data type is integer
namesListElements Character vector containing the names of the variables of which the data type is checked. Optional parameter, with as default value NULL. This argument should be used when the variable of which the data type is checked is not an object that was provided as an argument to the function, or when the list elements of the first argument do not have a name attached to it.

Value

No value is returned if all vectors have the integer data type. If not, an error message is thrown for each element of the list that does not pertain to the integer data type.

Examples

arg1 <- 1L
checkIntVec(list(arg1))

checkIntVec(list(1L, TRUE, 2L))

arg2 <- 'R'
checkIntVec(list(arg2))
checkIntVec(list(arg2, TRUE, 2))

checkLength checking if the length of the different elements of a list corresponds to what one expects.

Description

Checking if the length of the different elements of a list corresponds to what one expects.

Usage

checkLength(listObjects, lengthObjects)

Arguments

listObjects List of vectors, of irrespective data type.
lengthObjects Numeric vector, either of the same length as the 'listObjects' argument, or of length 1, but in the latter case, it will be tested whether or not the length of every element of the 'listObjects' argument equal this one value.

Value

No value is returned if all vectors correspond to the length against which it is tested. An error message is thrown when the length does not corresponds for at least one element of the list.
Examples

arg1 <- 'something'
checkLength(list(arg1), 1)

checkLength(list('somethingElse', TRUE), 1)
checkLength(list('somethingElse', TRUE), c(1, 1))

arg2 <- 2:5
checkLength(list(arg1, arg2), c(1, 4))
checkLength(list(arg1, arg2), 1)

checkLogicVec(listLogic, namesListElements = NULL)

Arguments

listLogic A list of the vectors of which one wishes to check if their data type is logical
namesListElements Character vector containing the names of the variables of which the data type is checked. Optional parameter, with as default value NULL. This argument should be used when the variable of which the data type is checked is not an object that was provided as an argument to the function, or when the list elements of the first argument do not have a name attached to it.

Value

No value is returned if all vectors have the logical data type. If not, an error message is thrown for each element of the list that does not pertain to the logical data type.

Examples

arg1 <- TRUE
checkLogicVec(list(arg1))

checkLogicVec(list(TRUE, T, 2))
checkLogicVec(list(TRUE, T, 2), c('Var1', 'Var2', 'Var3'))

arg2 <- 0.8
checkLogicVec(list(arg2))
checkLogicVec(list(arg2, 'T', 2))
checkNumOrIntVec

Checking if all elements of a list are all integer or numeric vectors

**Description**

Checking if all elements of a list are all integer or numeric vectors

**Usage**

```r
checkNumOrIntVec(listNumOrInt, namesListElements = NULL)
```

**Arguments**

- **listNumOrInt**: A list of the vectors of which one wishes to check if their data type is integer.
- **namesListElements**: Character vector containing the names of the variables of which the data type is checked. Optional parameter, with as default value NULL. This argument should be used when the variable of which the data type is checked is not an object that was provided as an argument to the function, or when the list elements of the first argument do not have a name attached to it.

**Value**

No value is returned if all vectors have the integer or numeric data type. If not, an error message is thrown for each element of the list that does not pertain to the integer or numeric data type.

**Examples**

```r
arg1 <- 1L
cHECKNUMORINTVEC(list(arg1))

arg1 <- 1
cHECKNUMORINTVEC(list(arg1))

CHECKNUMORINTVEC(list(1L, TRUE, 2L))
CHECKNUMORINTVEC(list(1L, TRUE, 2L), c('Var1', 'Var2', 'Var3'))

arg2 <- 'R'
cHECKNUMORINTVEC(list(arg2))
CHECKNUMORINTVEC(list(arg2, TRUE, 2))
```
checkNumVec  Checking if all elements of a list are all numeric vectors

Description
Checking if all elements of a list are all numeric vectors

Usage
checkNumVec(listNum, namesListElements = NULL)

Arguments

Arguments

- `listNum` A list of the vectors of which one wishes to check if their data type is numeric
- `namesListElements` Character vector containing the names of the variables of which the data type is checked. Optional parameter, with as default value NULL. This argument should be used when the variable of which the data type is checked is not an object that was provided as an argument to the function, or when the list elements of the first argument do not have a name attached to it.

Value
No value is returned if all vectors have the numeric data type. If not, an error message is thrown for each element of the list that does not pertain to the numeric data type.

Examples

```r
arg1 <- 2
checkNumVec(list(arg1))

checkNumVec(list(TRUE, T, 2))
checkNumVec(list(TRUE, T, 2), c('Var1', 'Var2', 'Var3'))

arg2 <- 0.8
checkNumVec(list(arg2))
cHECKNumVec(list(arg2, 'T', 2))
```

checkRanges  Checking if the value of a numeric or integer variable (of length 1) is located within a certain range.

Description
Checking if the value of a numeric or integer variable (of length 1) is located within a certain range.
Usage

checkRanges(listObjects, listRanges)

Arguments

listObjects  List of numeric or integer vectors, of each of length 1. It contains the list of variables of which one wants to test its value against a vector of valid values. This argument is obligatory.

listRanges  List of character vectors, each character vector should be of length 2 or 4, while the 'listRanges' list should be of the same length as the 'listObjects' argument. It contains the values against which one wants to test the 'listObjects' argument. This argument is obligatory.

Value

No value is returned if all vectors of the 'listObjects' argument is contained within the corresponding ranges of the 'listRanges' argument. An error message is thrown when this is not the case for at least one of the elements of the 'listObjects' argument. Note that each element of the 'listRange' argument should be of the following structure. The first element of the character vector, as well as the third element if the character vector is of length 4, should either be '>', '>=', '<' or '<='. In case that the length of the character vector is 4, the first and the third element should be opposite directions (some form of '>' combined with some form of '<'). The second and fourth element should be a numeric value coerced to a character. If the character vector is of length 2 (4), then the range is either bounded from below or (and) above.

Examples

someValue <- 2
checkRanges(list(someValue), list(c('<', 3)))

someValue <- '2'
checkRanges(list(someValue), list(c('<', 3)))
checkRanges(list(someValue), list(c(1.5, 3)))

someValue <- 6
someOtherValue <- 5
checkRanges(list(someValue, someOtherValue), list(c('>', 2.5), c('>=', 2.5, '<=', 5)))
checkRanges(list(someValue, someOtherValue), list(c('>', 2.5), c('>=', 2.5, '<', 5)))
checkRanges(list(someValue, someOtherValue), list(c('>', 2.5, '<=', 5), c('>', 2.5, '<', 5)))

checkValues  Checking if the value of vectors (of length 1) is authorized.

Description

Checking if the value of vectors (of length 1) is authorized.
Usage

checkValues(listObjects, listValues)

Arguments

listObjects        List of vectors, of irrespective data type and each of length 1. It contains the list
                   of variables of which one wants to test its value against a vector of valid values.
                   This argument is obligatory.

listValues        List of vectors, of irrespective data type and of the same length as the 'listO-
                   bjects' argument. It contains the values against which one wants to test the
                   'listObjects' argument. This argument is obligatory.

Value

No value is returned if all vectors correspond to the length against which it is tested. An error
message is thrown when at least one of the elements of the 'listObjects' contains an invalid value,
as stipulated by the 'listValues' argument.

Examples

lossType <- 'absolute'
checkValues(list(lossType), list(c('absolute', 'quadratic')))
checkValues(list(lossType), list(c('absolute', 'quadratic'), c('test', 'test2')))

#The next error message is weird, since it does not return the real name of the listObject
#that found to be wrong.
lossType <- 'absolute55'
listObjects <- list(lossType)
listValues <- list(c('absolute', 'quadratic'))
checkValues(listObjects, listValues)

#Now it is ok...
checkValues(list(lossType), list(c('absolute', 'quadratic')))
Index

checkCharVec, 2
checkIntVec, 2
checkLength, 3
checkLogicVec, 4
checkNumOrIntVec, 5
checkNumVec, 6
checkRanges, 6
checkValues, 7