Package 'OhdsiRTools'

December 23, 2016

Type Package

Title Tools for Maintaining OHDSI R Packages

Version 1.1.2

Date 2016-12-23

Author Martijn J. Schuemie [aut, cre],
Marc A. Suchard [aut],

Maintainer Martijn J. Schuemie <schuemie@ohdsi.org>

Description Format and check syntax of R code and packages following the OHDSI R style guidelines. Support for parallel computation.

License Apache License 2.0

Depends R (>= 3.1.0)

Imports codetools,
formatR,
snow,
RJSONIO,
RCurl,
XML,
jsonlite,
methods,
utils

Suggests testthat

NeedsCompilation no

RoxygenNote 5.0.1

R topics documented:

`checkUsagePackage` *Check all code in a package*

Description

Check all code in a package

Usage

```
checkUsagePackage(package, ignoreHiddenFunctions = TRUE,
  suppressBindingKeywords = c("ggplot2", "ffwhich", "subset.ffdf", "glm"))
```

Arguments

`package`	The name of the package to check.
`ignoreHiddenFunctions`	Ignore functions for which the definition cannot be retrieved?
`suppressBindingKeywords`	A set of keywords that are indicative of non-standard evaluation.

Details

This function uses the codetools package to check the code from problems. Heuristics are used to elimite false positives due to non-standard evaluation.

`clusterApply` *Apply a function to a list using the cluster*

Description

Apply a function to a list using the cluster

Usage

```
clusterApply(cluster, x, fun, ..., stopOnError = FALSE, progressBar = TRUE,
  divideFfMemory = TRUE, setFfTempDir = TRUE)
```

Arguments

cluster	The cluster of threads to run the function.
x	The list on which the function will be applied.
fun	The function to apply. Note that the context in which the function is specifies matters (see details).
...	Additional parameters for the function.
stopOnError	Stop when one of the threads reports an error? If FALSE, all errors will be reported at the end.
progressBar	Show a progress bar?
divideFfMemory	When TRUE, the memory available for processing ff and ffdf objects will be equally divided over the threads.
setFfTempDir	When TRUE, the ffTempDir option will be copied to each thread.

Details

The function will be executed on each element of x in the threads of the cluster. If there are more elements than threads, the elements will be queued. The progress bar will show the number of elements that have been completed.

It can sometimes be important to realize that the context in which a function is created is also transmitted to the worker node. If a function is defined inside another function, and that outer function is called with a large argument, that argument will be transmitted to the worker node each time the function is executed. It can therefore make sense to define the function to be called at the package level rather than inside a function, to save overhead.

Value

A list with the result of the function on each item in x.

---

clusterRequire *Require a package in the cluster*

Description

Require a package in the cluster

Usage

```
clusterRequire(cluster, package)
```

Arguments

cluster	The cluster object.
package	The name of the package to load in all nodes.

convertArgsToList *Deprecated: Convert arguments used in call to a list*

Description

Deprecated: Convert arguments used in call to a list

Usage

```
convertArgsToList(matchCall, resultClass = "list")
```

Arguments

matchCall	The result of match.call().
resultClass	The class of the resulting object.

Details

Takes the argument values (both default and user-specified) and store them in a list.

This function is deprecated because it fails when used in a function that is called using ::.

Value

An object of the class specified in resultClass.

Examples

```
myFun <- function(x = 1, y = 2) {
  return(convertArgsToList(match.call()))
}
```

createArgFunction *Create an argument function*

Description

Create an argument function

Usage

```
createArgFunction(functionName, excludeArgs = c(), includeArgs = NULL,
  addArgs = list(), rCode = c(), newName)
```

Arguments

functionName	The name of the function for which we want to create an args function.
excludeArgs	Exclude these arguments from appearing in the args function.
includeArgs	Include these arguments in the args function.
addArgs	Add these arguments to the args functions. Defined as a list with format name = default.
rCode	A character vector representing the R code where the new function should be appended to.
newName	The name of the new function. If not specified, the new name will be automatically derived from the old name.

Details

This function can be used to create a function that has (almost) the same interface as the specified function, and the output of this function will be a list of argument values.

Value

A character vector with the R code including the new function.

Examples

```
createArgFunction("read.csv", addArgs = list(exposureId = "exposureId"))
```

---

excludeFromList *Exclude variables from a list of objects of the same type*

Description

Exclude variables from a list of objects of the same type

Usage

```
excludeFromList(x, exclude)
```

Arguments

x	A list of objects of the same type.
exclude	A character vector of names of variables to exclude.

formatRFile *Format an R file*

Description

Format an R file

Usage

```
formatRFile(file, width.cutoff = 100)
```

Arguments

file	The path to the file.
width.cutoff	Number of characters that each line should be limited to.

formatRFolder *Format all R files in a folder*

Description

Format all R files in a folder

Usage

```
formatRFolder(path = ".", recursive = TRUE, skipAutogenerated = TRUE, ...)
```

Arguments

path	Path to the folder containing the files to format. Only files with the .R extension will be formatted.
recursive	Include all subfolders?
skipAutogenerated	Skip autogenerated files such as RcppExports.R?
...	Parameters to be passed on the the formatRFile function

Examples

```
## Not run:
formatRFolder()

## End(Not run)
```

formatRText *Format R code*

Description

Format R code

Usage

```
formatRText(text, width.cutoff = 100)
```

Arguments

text	A character vector with the R code to be formatted.
width.cutoff	Number of characters that each line should be limited to.

Value

A character vector with formatted R code.

insertCirceDefinitionInPackage *Load a Circe definition and insert it into this package*

Description

Load a Circe definition and insert it into this package

Usage

```
insertCirceDefinitionInPackage(definitionId, name = NULL,
  baseUrl = "http://hix.jnj.com:8080/WebAPI")
```

Arguments

definitionId	The number indicating which Circe definition to fetch.
name	The name that will be used for the json and SQL files. If not provided, the name in Circe will be used, but this may not lead to valid file names.
baseUrl	The base URL for the WebApi instance.

Details

Load a Circe definition from a WebApi instance and insert it into this package. This will fetch the json object and store it in the 'inst/circe' folder, and fetch the template SQL and store it in the 'inst/sql/sql_server' folder. Both folders will be created if they don't exist.

Examples

```
## Not run:
# This will create 'inst/circe/Angioedema.json' and
# 'inst/sql/sql_server/Angioedema.sql':

insertCirceDefinitionInPackage(2132, "Angioedema")

## End(Not run)
```

`loadSettingsFromJson` *Load a settings object from a JSON file*

Description

Load a settings object from a JSON file

Usage

```
loadSettingsFromJson(fileName)
```

Arguments

`fileName`	Name of the JSON file to load.

Details

Load a settings object from a JSON file, restoring object classes and attributes.

Value

An R object as specified by the JSON.

`makeCluster` *Create a cluster of nodes for parallel computation*

Description

Create a cluster of nodes for parallel computation

Usage

```
makeCluster(numberOfThreads, singleThreadToMain = TRUE)
```

Arguments

`numberOfThreads`	Number of parallel threads.
`singleThreadToMain`	If `numberOfThreads` is 1, should we fall back to running the process in the main thread?

Value

An object representing the cluster.

`matchInList` *In a list of object of the same type, find those that match the input*

Description

In a list of object of the same type, find those that match the input

Usage

```
matchInList(x, toMatch)
```

Arguments

`x`	A list of objects of the same type.
`toMatch`	The object to match.

Details

Typically, toMatch will contain a subset of the variables that are in the objects in the list. Any object matching all variables in `toMatch` will be included in the result.

Value

A list of objects that match the `toMatch` object.

`OhdsiRTools` *OhdsiRTools*

Description

OhdsiRTools

prettyPrint *Print a list of objects*

Description

Print a list of objects

Usage

```
prettyPrint(object)
```

Arguments

object	The list to print.

Details

Will print nested lists using indentation.

saveSettingsToJson *Save a settings object as JSON file*

Description

Save a settings object as JSON file

Usage

```
saveSettingsToJson(object, fileName)
```

Arguments

object	R object to be saved.
fileName	File name where the object should be saved.

Details

Save a setting object as a JSON file, using pretty formatting and preserving object classes and attributes.

selectFromList *Select variables from a list of objects of the same type*

Description

Select variables from a list of objects of the same type

Usage

```
selectFromList(x, select)
```

Arguments

x	A list of objects of the same type.
select	A character vector of names of variables to select.

stopCluster *Stop the cluster*

Description

Stop the cluster

Usage

```
stopCluster(cluster)
```

Arguments

cluster	The cluster to stop

updateCopyrightYearFile *Update the copyright year in a R or SQL file*

Description

Update the copyright year in a R or SQL file

Usage

```
updateCopyrightYearFile(file)
```

Arguments

file	The path to the file.

`updateCopyrightYearFolder` *Update the copyright year in all R and SQL files in a folder*

Description

Update the copyright year in all R and SQL files in a folder

Usage

```
updateCopyrightYearFolder(path = ".", recursive = TRUE)
```

Arguments

`path`	Path to the folder containing the files to update. Only files with the .R and .SQL extension will be updated.
`recursive`	Include all subfolders?

Examples

```
## Not run:
updateCopyrightYearFolder()

## End(Not run)
```

Index