

Package ‘OhdsiRTools’

August 28, 2017

Type Package

Title Tools for Maintaining OHDSI R Packages

Version 1.3.2

Date 2017-8-28

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 devtools,
codetools,
formatR,
snow,
RJSONIO,
RCurl,
XML,
jsonlite,
methods,
utils,
httr,
SqlRender,
stringr

Suggests testthat

NeedsCompilation no

RoxygenNote 6.0.1

R topics documented:

checkUsagePackage	2
clusterApply	3
clusterRequire	4
convertArgsToList	4
createArgFunction	5
excludeFromList	5

formatRFile	6
formatRFolder	6
formatRText	7
getCohortDefinitionName	7
getConceptSetConcepts	8
getConceptSetName	8
getPriorityVocabKey	9
insertCirceDefinitionInPackage	9
insertCohortDefinitionInPackage	10
insertCohortDefinitionSetInPackage	11
insertEnvironmentSnapshotInPackage	12
loadSettingsFromJson	12
makeCluster	13
matchInList	13
OhdsiRTools	14
prettyPrint	14
restoreEnvironment	14
saveSettingsToJson	15
selectFromList	15
stopCluster	16
takeEnvironmentSnapshot	16
updateCopyrightYearFile	17
updateCopyrightYearFolder	17

Index	18
--------------	-----------

checkUsagePackage	<i>Check all code in a package</i>
-------------------	------------------------------------

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 eliminate false positives due to non-standard evaluation.

clusterApply	<i>Apply a function to a list using the cluster</i>
--------------	---

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 specified 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	<i>Require a package in the cluster</i>
----------------	---

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	<i>Deprecated: Convert arguments used in call to a list</i>
-------------------	---

Description

Deprecated: Convert arguments used in call to a list

Usage

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

Arguments

matchCall	The result of <code>match.call()</code> .
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	<i>Format an R file</i>
-------------	-------------------------

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	<i>Format all R files in a folder</i>
---------------	---------------------------------------

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

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

Value

A character vector with formatted R code.

`getCohortDefinitionName`*Get a cohort definition's name from WebAPI*

Description

Get a cohort definition's name from WebAPI

Usage

```
getCohortDefinitionName(baseUrl, definitionId, formatName = FALSE)
```

Arguments

<code>baseUrl</code>	The base URL for the WebApi instance
<code>definitionId</code>	The cohort definition id in Atlas
<code>formatName</code>	Should the name be formatted to remove prefixes and underscores?

Details

Obtains the name of a cohort

Value

The name of the cohort

getConceptSetConcepts *Get Concept Set Concept Ids*

Description

Get Concept Set Concept Ids

Usage

```
getConceptSetConcepts(baseUrl, setId, vocabSourceKey = NULL)
```

Arguments

baseUrl	The base URL for the WebApi instance
setId	The concept set id in Atlas
vocabSourceKey	The source key of the Vocabulary. By default, the priority Vocabulary is used.

Details

Obtains the full list of concept Ids in a concept set

Value

A list of concept Ids

getConceptSetName *Get a concept set's name from WebAPI*

Description

Get a concept set's name from WebAPI

Usage

```
getConceptSetName(baseUrl, setId, formatName = FALSE)
```

Arguments

baseUrl	The base URL for the WebApi instance
setId	The concept set id in Atlas
formatName	Should the name be formatted to remove prefixes and underscores?

Details

Obtains the name of a concept set

Value

The name of the concept set

getPriorityVocabKey	<i>Get Priority Vocab Source Key</i>
---------------------	--------------------------------------

Description

Get Priority Vocab Source Key

Usage

```
getPriorityVocabKey(baseUrl)
```

Arguments

baseUrl	The base URL for the WebApi instance
---------	--------------------------------------

Details

Obtains the source key of the default OMOP Vocab in Atlas

Value

A string with the source key of the default OMOP Vocab in Atlas

insertCirceDefinitionInPackage	<i>Load a Circe definition and insert it into this package</i>
--------------------------------	--

Description

Load a Circe definition and insert it into this package

Usage

```
insertCirceDefinitionInPackage(definitionId, name = NULL, baseUrl)
```

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, for example: "http://api.ohdsi.org:80/WebAPI"

Details

Deprecated. Use [insertCohortDefinitionInPackage](#) instead.

insertCohortDefinitionInPackage

Load a cohort definition and insert it into this package

Description

Load a cohort definition and insert it into this package

Usage

```
insertCohortDefinitionInPackage(definitionId, name = NULL, baseUrl,
                                generateStats = FALSE, opts = list())
```

Arguments

definitionId	The number indicating which cohort definition to fetch.
name	The name that will be used for the json and SQL files. If not provided, the name in cohort will be used, but this may not lead to valid file names.
baseUrl	The base URL for the WebApi instance, for example: "http://api.ohdsi.org:80/WebAPI"
generateStats	Should the SQL include the code for generating inclusion rule statistics? Note that if TRUE, several additional tables are expected to exist as described in the details
opts	List of options that can be passed to the RCurl methods for specifying additional options for connecting to REST end-points

Details

Load a cohort definition from a WebApi instance and insert it into this package. This will fetch the json object and store it in the 'inst/cohorts' 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.

When using generateStats = TRUE, the following tables are required to exist when executing the SQL: cohort_inclusion, cohort_inclusion_result, cohort_inclusion_stats, and cohort_summary_stats. Also note that the cohort_inclusion table should be populated with the names of the rules prior to executing the cohort definition SQL.

Examples

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

insertCohortDefinitionInPackage(282, "Angioedema")

## End(Not run)
```

insertCohortDefinitionSetInPackage

Insert a set of cohort definitions into package

Description

Insert a set of cohort definitions into package

Usage

```
insertCohortDefinitionSetInPackage(fileName, baseUrl, insertTableSql = TRUE,
  insertCohortCreationR = TRUE, generateStats = FALSE, opts = list(),
  packageName)
```

Arguments

fileName	Name of a CSV file in the inst/settings folder of the package specifying the cohorts to insert. See details for the expected file format.
baseUrl	The base URL for the WebApi instance, for example: "http://api.ohdsi.org:80/WebAPI"
insertTableSql	Should the SQL for creating the cohort table be inserted into the package as well? This file will be called CreateCohortTable.sql.
insertCohortCreationR	Insert R code that will create the cohort table and instantiate the cohorts? This will create a file called R/CreateCohorts.R containing a function called .create-Cohorts.
generateStats	Should cohort inclusion rule statistics be created?
opts	List of options that can be passed to the RCurl methods for specifying additional options for connecting to REST end-points
packageName	The name of the package (only needed when inserting the R code as well).

Details

The CSV file should have at least the following fields:

atlasId The cohort ID in ATLAS.

cohortId The cohort ID that will be used when instantiating the cohort (can be different from atlasId).

name The name to be used for the cohort. This name will be used to generate file names, so please use letters and numbers only (no spaces).

```
insertEnvironmentSnapshotInPackage
```

Store snapshot of the R environment in the package

Description

Store snapshot of the R environment in the package

Usage

```
insertEnvironmentSnapshotInPackage(rootPackage)
```

Arguments

`rootPackage` The name of the root package

Details

This function records all versions used in the R environment that are used by one root package, and stores them in the R package that is currently being developed in a file called `inst/settings/rEnvironmentSnapshot.json`. This file can be used for example to restore the environment to the state it was when a particular study package was run using the [restoreEnvironment](#) function.

Examples

```
## Not run:  
insertEnvironmentSnapshotInPackage("OhdsiRTools")  
  
## 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	<i>OhdsiRTools</i>
-------------	--------------------

Description

OhdsiRTools

prettyPrint	<i>Print a list of objects</i>
-------------	--------------------------------

Description

Print a list of objects

Usage

```
prettyPrint(object)
```

Arguments

object The list to print.

Details

Will print nested lists using indentation.

restoreEnvironment	<i>Restore the R environment to a snapshot</i>
--------------------	--

Description

Restore the R environment to a snapshot

Usage

```
restoreEnvironment(snapshot, stopOnWrongRVersion = FALSE)
```

Arguments

snapshot The snapshot data frame as generated using the [takeEnvironmentSnapshot](#) function.

stopOnWrongRVersion Should the function stop when the wrong version of R is installed? Else just a warning will be thrown when the version doesn't match.

Details

This function restores the R environment to a previous snapshot, meaning all the packages will be restored to the versions they were at at the time of the snapshot. Note: on Windows you will very likely need to have RTools installed to build the various packages.

Examples

```
## Not run:
snapshot <- takeEnvironmentSnapshot("OhdsiRTools")
write.csv(snapshot, "snapshot.csv")

# 5 years later

snapshot <- read.csv("snapshot.csv")
restoreEnvironment(snapshot)

## End(Not run)
```

saveSettingsToJson	<i>Save a settings object as JSON file</i>
--------------------	--

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	<i>Select variables from a list of objects of the same type</i>
----------------	---

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	<i>Stop the cluster</i>
-------------	-------------------------

Description

Stop the cluster

Usage

```
stopCluster(cluster)
```

Arguments

cluster	The cluster to stop
---------	---------------------

takeEnvironmentSnapshot	<i>Take a snapshot of the R environment</i>
-------------------------	---

Description

Take a snapshot of the R environment

Usage

```
takeEnvironmentSnapshot(rootPackage)
```

Arguments

rootPackage	The name of the root package
-------------	------------------------------

Details

This function records all versions used in the R environment that are used by one root package. This can be used for example to restore the environment to the state it was when a particular study package was run using the [restoreEnvironment](#) function.

Value

A data frame listing all the dependencies of the root package and their version numbers, in the order in which they should be installed.

Examples

```
snapshot <- takeEnvironmentSnapshot("OhdsiRTools")
snapshot
```

`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

<code>file</code>	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

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

Examples

```
## Not run:  
updateCopyrightYearFolder()  
  
## End(Not run)
```

Index

checkUsagePackage, [2](#)
clusterApply, [3](#)
clusterRequire, [4](#)
convertArgsToList, [4](#)
createArgFunction, [5](#)

excludeFromList, [5](#)

formatRFile, [6](#)
formatRFolder, [6](#)
formatRText, [7](#)

getCohortDefinitionName, [7](#)
getConceptSetConcepts, [8](#)
getConceptSetName, [8](#)
getPriorityVocabKey, [9](#)

insertCirceDefinitionInPackage, [9](#)
insertCohortDefinitionInPackage, [9](#), [10](#)
insertCohortDefinitionSetInPackage, [11](#)
insertEnvironmentSnapshotInPackage, [12](#)

loadSettingsFromJson, [12](#)

makeCluster, [13](#)
matchInList, [13](#)

OhdsiRTools, [14](#)
OhdsiRTools-package (OhdsiRTools), [14](#)

prettyPrint, [14](#)

restoreEnvironment, [12](#), [14](#), [16](#)

saveSettingsToJson, [15](#)
selectFromList, [15](#)
stopCluster, [16](#)

takeEnvironmentSnapshot, [14](#), [16](#)

updateCopyrightYearFile, [17](#)
updateCopyrightYearFolder, [17](#)