Package ‘rapbase’

August 7, 2019

Type Package

Title Base Functions and Resources for Rapporteket

Version 1.10.0

Date 2019-07-23

Maintainer Are Edvardsen <biorakel@gmail.com>

Description Provide common functions and resources for registry specific R-packages at Rapporteket <https://rapporteket.github.io/rapporteket/articles/short_introduction.html>. This package is relevant for developers of packages/registries at Rapporteket.

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0)

Imports DBI, devtools, digest, gistr, httr, knitr, magrittr, readr, RJDBC, RMariaDB, sendmailR, shiny, utils, yaml

RoxygenNote 6.1.1

URL http://github.com/Rapporteket/rapbase

BugReports http://github.com/Rapporteket/rapbase/issues

Suggests testthat

NeedsCompilation no

Author Are Edvardsen [aut, cre] (<https://orcid.org/0000-0002-5210-3656>), Kevin Otto Thon [aut]

Repository CRAN

Date/Publication 2019-08-07 07:30:02 UTC

R topics documented:

`.getFun` *Provide explicit reference to function for do.call*

Description

Provide explicit reference to function for do.call

Usage

```
.getFun(x)
```

Arguments

`x`	string with explicit reference, i.e. 'package::function'

Value

value of the exported 'function' in 'package'

`.testAutoReport` *Simple test of automated report*

Description

Simple test of automated reporting from definitions provided in a yaml config file

Usage

```
.testAutoReport(aNum = 1, aChar = "a", anExp = Sys.Date())
```

Arguments

`aNum`	a number
`aChar`	a character
`anExp`	an expression

Value

A simple message listing the contents of the arguments

Examples

```
.testAutoReport()
```

appNavbarUserWidget *Create widget for registry apps at Rapporteket*

Description

Provides a widget-like information and utility block to be applied to all registry apps at Rapporteket. Contains the user name, organization and logout/exit as hyperlinked text.

Usage

```
appNavbarUserWidget(user = "Undefined person",
  organization = "Undefined organization")
```

Arguments

`user`	String providing the name of the user
`organization`	String providing the organization of the user

Details

Normally, user information will be be provided through the session parameter and hence this will have to be provided from the server. The "rendering" of this info must hence be done within a layout element at the client such as a `tabPanel`. Selecting any one of them should be fine... At the client, both `uiOutput` and `textOutput` will be fine "rendering the information provided by the server.

Example of use in shiny (pseudo code):

```
server <- function(input, output, session) {
  ...
  output$appUserName <- renderText(getUserName(session))
  output$appUserOrg <- renderText(getUserReshId(session))
  ...
}

ui <- tagList(
  navbarPage(
    ...,
    tabPanel(...,
   appNavbarUserWidget(user = uiOutput(appUserName), organization = textOutput(appUserOrg))
    ),
    ...
  )
)
```

Value

Ready made html script

Examples

```
appNavbarUserWidget()
```

createAutoReport *Create and add report to config*

Description

Adds an entry to the system configuration of reports to run at given intervals. After generating the configuration from the new entry the function load the current system configuration, adds the new entry and saves the updated system configuration.

Usage

```
createAutoReport(synopsis, package, fun, paramNames, paramValues, owner,
  email, organization, runDayOfYear, dryRun = FALSE)
```

Arguments

synopsis	String with description of the report and to be used in subject field of email distributed reports
package	String with package name also corresponding to registry
fun	String providing name of function to be called for generating report
paramNames	String vector where each element corresponds to the input parameter to be used in the above function
paramValues	String vector with corresponding values to paramNames
owner	String providing the owner of the report. Usually a user name
email	String with email address to recipient of email containing the report
organization	String identifying the organization the owner belongs to
runDayOfYear	Integer vector with day numbers of the year when the report is to be run
dryRun	Logical defining if global auto report config actually is to be updated. If set to TRUE the actual config (all of it) will be returned by the function. FALSE by default

Value

Nothing unless dryRun is set TRUE in which case a list of all config will be returned

See Also

deleteAutoReport

deleteAutoReport *Delete existing report from config*

Description

Delete existing report from config

Usage

```
deleteAutoReport(autoReportId)
```

Arguments

`autoReportId`	String providing the auto report unique id

See Also

`createAutoReport`

findNextRunDate *Find next run date for automated reports*

Description

Find the next date that an automated report is supposed to be run. Likely, this function will only be relevant for shiny apps when this date is to be printed

Usage

```
findNextRunDate(runDayOfYear, baseDayNum = as.POSIXlt(Sys.Date())$yday +
  1, returnFormat = "%A %d. %B %Y")
```

Arguments

`runDayOfYear`	Numeric vector providing year-day numbers
`baseDayNum`	Numeric defining base year-day. Default is today
`returnFormat`	String providing return format as described in `strptime` in the current locale. Defaults to "%A %d. %B %Y" that will provide something like 'Mandag 20. januar 2019' in a Norwegian locale

Value

String date for printing

Examples

```
# Will return Jan 30 in the current year and locale with default formatting
findNextRunDate(c(10, 20, 30), 20)
```

getConfig *Get configuration for package, if any*

Description

Try to obtain yaml-formatted configuration placed either as given by the environment variable R_RAP_CONFIG_PATH or as provided by the package itself. If none can be found the function exits with an error

Usage

```
getConfig(fileName = "dbConfig.yml", packageName = "rapbase")
```

Arguments

fileName	String providing configuration file base name
packageName	String providing the package name

Value

A list of (yaml) configuration

Examples

```
getConfig()
```

getRapPackages *Get all installed Rapporteket packages*

Description

Get all installed packages that depends on 'rapbase' which itself will not be reported

Usage

```
getRapPackages()
```

Value

Character vector of packages names

Examples

```
getRapPackages()
```

getRegs *Provide vector of registries (*i.e.* their R packages) in config*

Description

Provide vector of registries (*i.e.* their R packages) in config

Usage

```
getRegs(config)
```

Arguments

`config`	list of configuration for automated reports

Value

character vector of registry (package) names

getShinyUserGroups *Get user groups from a shiny session object*

Description

Get user groups from a shiny session object

Usage

```
getShinyUserGroups(shinySession, testCase = FALSE)
```

Arguments

`shinySession`	List shiny session object
`testCase`	Logical defining to use shinySessionInfo on a test case. For test cases user privileges are provided by a query defined in the url of the shiny application request. Default FALSE

Value

String user groups

See Also

`getShinyUserName`, `getShinyUserReshId`, `getShinyUserRole`

Examples

```
# Requires a valid shiny session object
getShinyUserGroups(shinySessionObject)
```

`getShinyUserName` *Get user name from a shiny session object*

Description

Get user name from a shiny session object

Usage

```
getShinyUserName(shinySession, testCase = FALSE)
```

Arguments

`shinySession`	List shiny session object
`testCase`	Logical defining to use shinySessionInfo on a test case. For test cases user privileges are provided by a query defined in the url of the shiny application request. Default FALSE

Value

String user name

See Also

`getShinyUserName`, `getShinyUserGroups`, `getShinyUserReshId`, `getShinyUserRole`

Examples

```
# Requires a valid shiny session object
getShinyUserName(shinySessionObject)
```

getShinyUserReshId *Get user resh_id from a shiny session object*

Description

Get user resh_id from a shiny session object

Usage

```
getShinyUserReshId(shinySession, testCase = FALSE)
```

Arguments

shinySession	List shiny session object
testCase	Logical defining to use shinySessionInfo on a test case. For test cases user privileges are provided by a query defined in the url of the shiny application request. Default FALSE

Value

String user resh_id

See Also

getShinyUserName, getShinyUserGroups, getShinyUserRole

Examples

```
# Requires a valid shiny session object
getShinyUserReshId(shinySessionObject)
```

getShinyUserRole *Get user role from a shiny session object*

Description

Get user role from a shiny session object

Usage

```
getShinyUserRole(shinySession, testCase = FALSE)
```

Arguments

`shinySession` List shiny session object

`testCase` Logical defining to use shinySessionInfo on a test case. For test cases user privileges are provided by a query defined in the url of the shiny application request. Default FALSE

Value

String user role

See Also

`getShinyUserName`, `getShinyUserGroups`, `getShinyUserReshId`

Examples

```
# Requires a valid shiny session object
getShinyUserRole(shinySessionObject)
```

`getUserEmail` *Get user email from config or session object*

Description

This is a helper function for `userInfo`. When used without a shiny session object calls to this function is made without any arguments. If redefining contexts is needed, please use `userInfo` instead.

Usage

```
getUserEmail(shinySession = NULL)
```

Arguments

`shinySession` Shiny session object (list, NULL by default). Must be provided when the source of user attributes is either the shiny app url or an external authentication provider. By default this will apply to the ’TEST’, ’QA’ and ’PRODUCTION’ contexts in which case the shiny session object must be provided.

Value

String email address

See Also

getUserName, getUserGroups, getUserReshId, getUserEmail

Examples

```
# Requires a valid shiny session object
getUserEmail()
getUserEmail(shinySessionObject)
```

getUserFullName *Get user full name from config or session object*

Description

This is a helper function for userInfo. When used without a shiny session object calls to this function is made without any arguments. If redefining contexts is needed, please use userInfo instead.

Usage

```
getUserFullName(shinySession = NULL)
```

Arguments

shinySession	Shiny session object (list, NULL by default). Must be provided when the source of user attributes is either the shiny app url or an external authentication provider. By default this will apply to the 'TEST', 'QA' and 'PRODUCTION' contexts in which case the shiny session object must be provided.

Value

String full name

See Also

getUserName, getUserGroups, getUserReshId, getUserEmail, getUserPhone

Examples

```
# Requires a valid shiny session object
getUserFullName()
getUserFullName(shinySessionObject)
```

getUserGroups *Get user groups from config or session object*

Description

This is a helper function for userInfo. When used without a shiny session object calls to this function is made without any arguments. If redefining contexts is needed, please use userInfo instead.

Usage

```
getUserGroups(shinySession = NULL)
```

Arguments

shinySession	Shiny session object (list, NULL by default). Must be provided when the source of user attributes is either the shiny app url or an external authentication provider. By default this will apply to the 'TEST', 'QA' and 'PRODUCTION' contexts in which case the shiny session object must be provided.

Value

String user name

See Also

getUserName, getUserReshId, getUserRole

Examples

```
# Requires a valid shiny session object
getUserGroups()
getUserGroups(shinySessionObject)
```

getUserName *Get user name from config or session object*

Description

This is a helper function for userInfo. When used without a shiny session object calls to this function is made without any arguments. If redefining contexts is needed, please use userInfo instead.

Usage

```
getUserName(shinySession = NULL)
```

Arguments

`shinySession`	Shiny session object (list, NULL by default). Must be provided when the source of user attributes is either the shiny app url or an external authentication provider. By default this will apply to the 'TEST', 'QA' and 'PRODUCTION' contexts in which case the shiny session object must be provided.

Value

String user name

See Also

`getUserGroups`, `getUserReshId`, `getUserRole`

Examples

```
# Requires a valid shiny session object
getUserName()
getUserName(shinySessionObject)
```

`getUserPhone` *Get user phone (number) from config or session object*

Description

This is a helper function for `userInfo`. When used without a shiny session object calls to this function is made without any arguments. If redefining contexts is needed, please use `userInfo` instead.

Usage

```
getUserPhone(shinySession = NULL)
```

Arguments

`shinySession`	Shiny session object (list, NULL by default). Must be provided when the source of user attributes is either the shiny app url or an external authentication provider. By default this will apply to the 'TEST', 'QA' and 'PRODUCTION' contexts in which case the shiny session object must be provided.

Value

String phone number

See Also

getUserName, getUserGroups, getUserReshId, getUserEmail, getUserFullName

Examples

```
# Requires a valid shiny session object
getUserPhone()
getUserPhone(shinySessionObject)
```

getUserReshId *Get user resh ID from config or session object*

Description

This is a helper function for userInfo. When used without a shiny session object calls to this function is made without any arguments. If redefining contexts is needed, please use userInfo instead.

Usage

```
getUserReshId(shinySession = NULL)
```

Arguments

shinySession	Shiny session object (list, NULL by default). Must be provided when the source of user attributes is either the shiny app url or an external authentication provider. By default this will apply to the 'TEST', 'QA' and 'PRODUCTION' contexts in which case the shiny session object must be provided.

Value

String user name

See Also

getUserName, getUserGroups, getUserRole

Examples

```
# Requires a valid shiny session object
getUserReshId()
getUserReshId(shinySessionObject)
```

---

getUserRole *Get user role from config or session object*

Description

This is a helper function for userInfo. When used without a shiny session object calls to this function is made without any arguments. If redefining contexts is needed, please use userInfo instead.

Usage

```
getUserRole(shinySession = NULL)
```

Arguments

shinySession	Shiny session object (list, NULL by default). Must be provided when the source of user attributes is either the shiny app url or an external authentication provider. By default this will apply to the 'TEST', 'QA' and 'PRODUCTION' contexts in which case the shiny session object must be provided.

Value

String user name

See Also

getUserName, getUserGroups, getUserReshId

Examples

```
# Requires a valid shiny session object
getUserRole()
getUserRole(shinySessionObject)
```

`HalloRapporteket` *Plain testing tool*

Description

To be used for testing purposes

Usage

```
HalloRapporteket()
```

Value

message A test message

`Helligdager2008til2022` *Offisielle norske helligdager 2008 til 2022*

Description

Et datasett med norske røddager 2008 til 2022. Søndager er ikke inkludert i listen.

Usage

```
Helligdager2008til2022
```

Format

En data frame med 195 rader og 2 kolonner

Dato Dato helligdagen faller på, i datoformat

Helligdag Hvilken helligdag det gjelder

Source

`http://www.norskkalender.no/`

installGithubPackage *Install packages from GitHub at Rapporteket*

Description

Install and report results

Usage

```
installGithubPackage(packageName, branchName, readConfig = TRUE)
```

Arguments

packageName	String Name of the package
branchName	String Name of the branch to use
readConfig	Set to FALSE to prevent function from reading configuration. Set TRUE by default. Mainly used for testing purposes.

Value

story String containing logged entries from the function

isPkgRapReg *Test if a package is part of Rapporteket*

Description

Test if an installed package is linked to Rapportekt based on dependency to the package ’rapbase’

Usage

```
isPkgRapReg(pkg)
```

Arguments

pkg	String providing the package name

Value

Logical TRUE if pkg depends on ’rapbase’, FALSE if not

See Also

`getRapPackages` on how to list all packages that depend om ’rapbase’

Examples

```
# returns FALSE, rapbase has no explicit dependency to itself
isPkgRapReg("rapbase")
```

`LoadRegData` *Provider of data for registries at Rapporteket*

Description

Generic to registries, provide reporting data obtained from sql databases Underlying this function is rapbase::RapporteketDbConnection

Usage

```
LoadRegData(registryName, query, dbType = "mysql")
```

Arguments

`registryName` String Name of the registry as defined in dbConfig.yml

`query` String SQL query to obtain the data

`dbType` String Type of db to query, currently "mysql" (default) and "mssql"

Value

RegData dataframe Registry data

`MakeMessage` *Provide simple formatting of a sequential message*

Description

Wrap strings into a message for simple presentation

Usage

```
MakeMessage(story, message, newLine = TRUE, newPara = FALSE,
  timeStamp = TRUE)
```

Arguments

`story` String Text that the message is to be added to

`message` String Message to add

`newLine` Logical [TRUE] Start message at a new line. TRUE by default

`newPara` Logical [FALSE] Start message at new paragraph. Default FALSE

`timeStamp` Logical [TRUE] Add a time stamp to the message. TRUE by default

Value

story String The whole story...

`makeRunDayOfYearSequence`

Make a sequence of day numbers from av given date and interval

---

Description

This function provides an even sequence of day numbers spanning 365/366 days from the start date and interval provided. Mainly to be used in setting up automated reports at Rapporteket

Usage

```
makeRunDayOfYearSequence(startDay = Sys.Date(), interval)
```

Arguments

`startDay`	Start date of sequence. May be provided as a string, *e.g.* \"2019-03-17\" or as class \"Date\". Defaults to today
`interval`	String representing a valid seq.POSIXt interval such as "DSTday", "week", "month", "quarter" or "year")

Value

Integer vector of day numbers

Examples

```
makeRunDayOfYearSequence(interval = "month")
```

`makeUserSubscriptionTab`

Make table of subscriptions of reports

Description

Make a table to be rendered in a shiny app providing the active subscriptions of a given user within a given registry which are both collected from the shiny session object provided

Usage

```
makeUserSubscriptionTab(session)
```

Arguments

session A shiny session object

Value

Matrix providing a table to be rendered in a shiny app

PushGist *Push some markdown to gist*

Description

Based on Rmd documents, push the resulting md to defined gist

Usage

```
PushGist(mdFile, githubUserName = "")
```

Arguments

mdFile String absolute file name and path of source Rmd-file. For instance provided as a system.file expression.

githubUserName String github user name for whom the gist will be published

rapbase *rapbase: Base Functions and Resources for Rapporteket*

Description

Provide common functions and resources for registry specific R-packages at Rapporteket. This packages is relevant for developers of packages/registries at Rapporteket

rapCloseDbConnection *Close down data connection handle*

Description

Close down data connection handle

Usage

```
rapCloseDbConnection(con)
```

Arguments

con Open connection object that is to be closed

rapOpenDbConnection *Provide connection handle for data source at Rapporteket*

Description

Generic to registries, handle the data source connections, including usernames and passwords needed to open these connections

Usage

```
rapOpenDbConnection(registryName, dbType = "mysql")
```

Arguments

registryName	String id used for the registry in global configuration file from which information on the database connection is provided
dbType	String providing type of data source, one of "mysql" and "mssql". Defaults to "mysql"

Value

con Data source connection object

drv DBIDriver object

readAutoReportData *Read automated report metadata*

Description

Read automated report metadata

Usage

```
readAutoReportData(fileName = "autoReport.yml",
  packageName = "rapbase")
```

Arguments

fileName	String defining name of the yaml configuration file. Default 'autoReport.yml'
packageName	String defining the package in which the above configuration file resides. A configuration file within an R-package is only used in case the environmental variable 'R_RAP_CONFIG_PATH' is not defined (empty)

Value

a list of yaml data

Examples

```
readAutoReportData()
```

---

runAutoReport *Run reports as defined in yaml config and ship content by email*

Description

Usually to be called by a scheduler, e.g. cron. If the provided day of year matches those of the config the report is run as otherwise specified in config. Functions called upon are expected to return a path to a file that can be attached to an email. The email itself is defined and sent to recipients defined in the config

Usage

```
runAutoReport(dayNumber = as.POSIXlt(Sys.Date())$yday + 1,
  dryRun = FALSE)
```

Arguments

dayNumber	Integer day of year where January 1st is 1. Defaults to current day, i.e. as.POSIXlt(Sys.Date())$yday+1 (POSIXlt yday is base 0)
dryRun	Logical defining if emails are to be sent. If TRUE a message with reference to the payload file is given but no emails will actually be sent. Default is FALSE

Value

Emails with corresponding file attachment. If dryRun == TRUE just a message

Examples

```
# Example depend on environment variable R_RAP_CONFIG_PATH being set
runAutoReport()
```

`RunNoweb` *RunNoweb*

Description

Function to run noweb file contained in a package. Assume all noweb files of the package are placed flat under the *inst* directory

Usage

```
RunNoweb(nowebFileName, packageName, weaveMethod = "knitr")
```

Arguments

`nowebFileName` Basename of the noweb file, *e.g.* 'myFile.Rnw'.

`packageName` Name of the package containing noweb file(s)

`weaveMethod` Method to apply for weaving. Currently available are 'Sweave' and 'knitr', default to the latter.

`runShinyApp` *runShinyApp starts a shiny application in a package*

Description

runShinyApp start a shiny application residing within an R package. First attempt is to keep this function generic (across packages). It might not prove wise and future implementation might be per-package.

Usage

```
runShinyApp(appName, appsDirectoryName, packageName)
```

Arguments

`appName` String providing the name of the shiny application

`appsDirectoryName`
String providing the within-package-path to its shiny application(s). Assumingly, in the source tree for a given R package an app will be found under *inst/shinyApps/anApp*. In this case, *appsDirectoryName* must be set to *'shinyApps'*.

`packageName` String providing the name of the package in which the shiny app resides

Details

Idea and code (partly) taken from Dean Attali's R Blog. Gratitude.

`selectByOrganization` *Select data on one organization from config (list)*

Description

Pick all config corresponding to a given organization (of the report)

Usage

```
selectByOrganization(config, organization)
```

Arguments

`config`	list of configuration for automated reports
`organization`	string giving the exact organization

Value

list with config for reports belonging to organization

`selectByOwner` *Select data on one owner from config (list)*

Description

Pick all config corresponding to a given owner (of the report)

Usage

```
selectByOwner(config, owner)
```

Arguments

`config`	list of configuration for automated reports
`owner`	string giving the exact name owner

Value

list with config for reports belonging to owner

`selectByReg` *Select data on one registry from config (list)*

Description

Pick all config corresponding to a given registry. Registry name is not given as such, but rather as its corresponding R package name. Hence, a registry must be given as the name of its R package

Usage

```
selectByReg(config, reg)
```

Arguments

`config`	list of configuration for automated reports
`reg`	string giving the exact name of the R package for the registry

Value

list with config for registry reg

`shinySessionInfo` *Provide elements from a shiny session object*

Description

Extracts elements from a shiny session object relevant for user data. This function will normally be used via its helper functions.

Usage

```
shinySessionInfo(shinySession, entity, testCase = FALSE)
```

Arguments

`shinySession`	List shiny session object
`entity`	String defining the element to return. One of 'user', 'groups', 'resh_id' or 'role'
`testCase`	Logical defining to use shinySessionInfo on a test case. For test cases user privileges are provided by a query defined in the url of the shiny application request. Default FALSE

Value

String of single user data element

See Also

getShinyUserName, getShinyUserGroups, getShinyUserReshId, getShinyUserRole

---

userInfo *Provide user attributes based on environment context*

Description

Extracts elements from either config, url (shiny) or session (shiny) relevant for user data such as name, group, role and reshId. Source of info is based on environment context and can be controlled by altering the default settings for which contexts that will apply for the various sources of user data. This function will normally be used via its helper functions (see below).

Usage

```
userInfo(entity, shinySession = NULL, devContexts = c("DEV"),
  testContexts = c("TEST"), prodContexts = c("QA", "PRODUCTION"))
```

Arguments

entity	String defining the element to return. Currently, one of 'user', groups', 'resh_id', 'role', 'email', 'full_name' or 'phone'
shinySession	Shiny session object (list, NULL by default). Must be provided when the source of user attributes is either the shiny app url or an external authentication provider. By default this will apply to the 'TEST', 'QA' and 'PRODUCTION' contexts in which case the shiny session object must be provided.
devContexts	A character vector providing unique instances to be regarded as a development context. In this context user attributes will be read from configuration as provided by 'rapbaseConfig.yml'. The instances provided cannot overlap instances provided in any other contexts. By default set to `c("DEV")`.
testContexts	A character vector providing unique instances to be regarded as a test context. In this context user attributes will be read from the url call to a shiny application. Hence, for this context the corresponding shiny session object must also be provided. The instances provided cannot overlap instances provided in any other contexts. By default set to `c("TEST")`.
prodContexts	A character vector providing unique instances to be regarded as a production context. In this context user attributes will be read from the shiny session object (as shiny server interacts with an external log-in service). Hence, for this context the corresponding shiny session object must also be provided. The instances provided cannot overlap instances provided in any other contexts. By default set to `c("QA","PRODUCTION")`.

Value

String of single user data element

See Also

getUserName, getUserGroups, getUserReshId, getUserRole

writeAutoReportData *Write automated report metadata*

Description

Write automated report metadata

Usage

```
writeAutoReportData(fileName = "autoReport.yml", config,
  packageName = "rapbase")
```

Arguments

fileName	String defining name of the yaml configuration file. Default 'autoReport.yml'
config	a list of yaml configuration
packageName	String defining the package in which the above configuration file resides. A configuration file within an R-package is only used in case the environmental variable 'R_RAP_CONFIG_PATH' is not defined (empty)

Examples

```
# Example depend on environment variable R_RAP_CONFIG_PATH being set
config <- readAutoReportData()
writeAutoReportData(config = config)
```

Index