

Package: SurveySolutionsAPIv2 (via r-universe)

October 24, 2026

Title Comprehensive API Access to Your 'Survey Solutions' Server Using 'httr2'

Version 0.1.2

Description A comprehensive suite of functions for interfacing with the 'Survey Solutions' APIs (<https://demo.mysurvey.solutions/apidocs/index.html> and <https://docs.mysurvey.solutions/>). It supports all the current POST, GET, and PATCH requests of the 'REST' API and extends its capabilities by integrating selected 'GraphQL' API queries and mutations when 'REST' API equivalents are not available. Designed for ease of use, it includes several convenience functions to streamline operations. The package ensures that data retrieved from the API is automatically transformed into a single, easy-to-use data frame or data table, optimizing it for subsequent analysis and processing. Additionally, it is specifically designed to facilitate seamless integration into 'shiny' applications, enhancing the interactive web application development experience. Built on the robust 'httr2' package, this version supersedes the previous 'httr'-based implementation. Recognizing the needs of large-scale data operations such as censuses, the package also incorporates features for parallel processing, enhancing its efficiency and scalability.

License MIT + file LICENSE

URL <https://michael-cw.github.io/SurveySolutionsAPIv2/>,
<https://github.com/michael-cw/SurveySolutionsAPIv2>

BugReports <https://github.com/michael-cw/SurveySolutionsAPIv2/issues>

Depends R (>= 4.1.0)

Imports cli, curl, data.table, dplyr, foreach, graphics, httr2, jsonlite, lubridate, readr, rlang, shiny, stats, stringr, susographql (>= 0.1.1), tidyjson, tidyr, withr, zip

Encoding UTF-8
Language en
RoxygenNote 7.3.2
Config/testthat/edition 3
Suggests doFuture, DT, future, future.apply, ggplot2, knitr,
 progressr, readxl, rmarkdown, sf, testthat (>= 3.0.0)
VignetteBuilder knitr
NeedsCompilation no
Author Michael Wild [aut, cre, cph]
Maintainer Michael Wild <michael.wild@me.com>
Repository <https://cran.r-universe.dev>
Date/Publication 2026-10-24 16:20:13 UTC
RemoteUrl <https://github.com/cran/SurveySolutionsAPIv2>
RemoteRef HEAD
RemoteSha 45e228dce3810c757d4b146e8d4895cb112f7bdc

Contents

assignmentClass	4
boxplot_summary	4
calendarevent	5
exportClass	7
getinfo	8
is.assignmentClass	9
is.exportClass	10
is.UserClass	10
summaryTable	11
summaryTable.UserClass	13
suso.options	14
suso_archUSR	15
suso_assignInterview	16
suso_assignWorkspace	17
suso_cancelExport	19
suso_clear_keys	20
suso_commentInterview	21
suso_createASS	22
suso_createUSER	23
suso_createWorkspace	24
suso_deleteInterview	26
suso_deletemap	27
suso_deleteWorkspace	28
suso_enableWorkspace	28
suso_export	29
suso_export_paradata	32

suso_get_api_key	35
suso_get_assignments	36
suso_get_default_key	38
suso_get_stats	39
suso_get_stats_interview	40
suso_getAllAnswerInterview	41
suso_getAllHistoryInterview	42
suso_getAllInterviewQuestionnaire	43
suso_getExportList	44
suso_getExportProcess	46
suso_getINT	47
suso_getINT_info	48
suso_getInterviewPDF	49
suso_getQuestDetails	50
suso_getQuestionsQuestionnaire	52
suso_getStatsQuestionnaires	53
suso_getSV	54
suso_getSV_info	55
suso_getUSR	56
suso_getWorkspace	57
suso_globalNotice	58
suso_keys	59
suso_mapassign	60
suso_mapinfo	61
suso_mapreport	62
suso_mapupload	64
suso_patchApproveInterview	65
suso_patchRejectInterview	66
suso_PwCheck	68
suso_questCriticalityLevel	69
suso_questRecordAudio	70
suso_set_assignments	71
suso_set_key	72
suso_set_maxpar_req	74
suso_set_pwcheck_mess	74
suso_set_workspace	75
suso_updateWorkspace	76
UserClass	77

assignmentClass	<i>Assignment Classes & Methods</i>
-----------------	---

Description

Assignments class extends the data.table class to include relevant methods

Usage

```
assignmentClass(x, ...)
```

Arguments

x	list returned by api call
...	additional attributes to be added to the assignmentClass

Details

The assignmentClass is a data.table extended with additional attributes and methods, depending on the api endpoint. The assignmentClass is used as input for the getInfo function.

Value

assignmentClass object

Examples

```
ass_list <- list(Assignments = data.frame(Id = 1, ResponsibleName = "interviewer1"))
ac <- assignmentClass(ass_list)
is.assignmentClass(ac)
```

boxplot_summary	<i>S3 method to create a box plot of numeric variables for an exportClass object</i>
-----------------	--

Description

This function generates a box plot for the numeric variables in a exportClass object.

Usage

```
boxplot_summary(x, useGGplot2 = FALSE, ...)

## S3 method for class 'exportClass'
boxplot_summary(x, useGGplot2 = FALSE, ...)
```

Arguments

<code>x</code>	A summaryTable object.
<code>useGGplot2</code>	Logical, if TRUE (default) the function returns a ggplot2 object, otherwise a base R boxplot is returned.
<code>...</code>	Additional arguments (not used).

Value

A box plot visualizing the numeric variables.

A ggplot2 plot object if useGGplot2 is TRUE, or a base R boxplot list invisibly otherwise

Examples

```
dt <- data.table::data.table(q1 = 1:10, q2 = 11:20)
vlabs <- data.table::data.table(
  VariableName = c("q1", "q2"),
  QuestionText = c("Question 1", "Question 2")
)
ec <- exportClass(dt, varLabels = vlabs)
boxplot_summary(ec)
```

calendarevent

Create, update or delete a calendar event

Description

Create, update or delete a calendar event for one or several assignments or interviews.

Usage

```
suso_createCALEV(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  workspace = suso_get_api_key("workspace"),
  token = NULL,
  intID = NULL,
  AssId = NULL,
  comment = NULL,
  startDate = NULL,
  startTime = NULL,
  startTZ = "UTC"
)

suso_updateCALEV(
```

```

    server = suso_get_api_key("susoserver"),
    apiUser = suso_get_api_key("susouser"),
    apiPass = suso_get_api_key("susopass"),
    workspace = suso_get_api_key("workspace"),
    token = NULL,
    publicKey = NULL,
    comment = NULL,
    startDate = NULL,
    startTime = NULL,
    startTZ = "UTC"
)

suso_delCALEV(
  server = suso_get_api_key("susoserver"),
  apiUser = suso_get_api_key("susouser"),
  apiPass = suso_get_api_key("susopass"),
  workspace = suso_get_api_key("workspace"),
  token = NULL,
  publicKey = NULL
)

```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
intID	UUID vector of interview id(s)
AssId	numeric vector of assignment id(s)
comment	a comment string
startDate	new start date, format must be: 2024-01-16
startTime	new start date, format must be: 01:41:14
startTZ	time zone of the tablet device, use OlsonNames
publicKey	UUID vector of calendar event id(s)

Details

This function creates a calendar event either for one or several assignments or one or several interviews. Either one of *AssID* or *intID* must not be *NULL*. If *length(AssID)* or *length(intID)* is greater 1, then *startDate*, *startTime* and *startTZ* must either be each of length 1 (same event for all) or of the same length, allowing for multiple events with different times.

Value

a data.table with the created event(s).

Functions

- suso_createCALEV(): add a calendar event
- suso_updateCALEV(): update a calendar event
- suso_delCALEV(): delete a calendar event

Examples

```
# add single calendar event to assignment
calEvass<-suso_createCALEV(AssId = 268,
                           startDate = "2024-01-17",
                           startTime = "10:40:00")

# update single event
calEvup<-suso_updateCALEV(publicKey = calEvass$publicKey[1],
                           startDate = "2024-01-18",
                           startTime = "10:40:00")

# delete single event
calEvDel<-suso_delCALEV(publicKey = calEvup$publicKey[1])

# add calendar event to multiple assignments
calEvass<-suso_createCALEV(AssId = 55:60,
                           startDate = "2024-01-18",
                           startTime = "11:40:00")

# delete multiple calendar events
calEvDel<-suso_delCALEV(publicKey = calEvass$publicKey)
```

exportClass	<i>Export Classes & Methods</i>
-------------	-------------------------------------

Description

Export class extends the data.table class to include relevant methods

Usage

```
exportClass(x, varLabels, ..., type = "main")
```

Arguments

x	list returned by api call
varLabels	data.table of variable labels as returned by suso_getQuestDetails(operation.type = "structure")
...	additional attributes to be added to the ExportClass
type	one of main or para, if main returns exportClass object for the main data, if para, returns an exportClass object for the paradata.

Value

An object of class `exportClass` (inheriting from `data.table`)

Examples

```
dt <- data.table::data.table(q1 = 1:5)
vlabs <- data.table::data.table(VariableName = "q1", QuestionText = "Question 1")
ec <- exportClass(dt, varLabels = vlabs)
is.exportClass(ec)
```

getinfo

assignmentClass methods

Description

`getinfo` allows you to retrieve relevant additional information from the `assignmentClass`, object depending on the api endpoint.

Usage

```
getinfo(obj, arg)

## S3 method for class 'assignmentClass'
getinfo(obj, arg)

## S3 method for class 'exportClass'
getinfo(obj, arg)

## S3 method for class 'UserClass'
getinfo(obj, arg)
```

Arguments

<code>obj</code>	object of <code>assignmentClass</code>
<code>arg</code>	name of attribute, if <code>arg="arglist"</code> then it returns all available arguments

Details

To retrieve all available arguments use `arg="arglist"`

Value

the specific attribute

Examples

```
ass_list <- list(
  Assignments = data.frame(Id = 1, ResponsibleName = "interviewer1"),
  ResponsibleId = "a1b2c3d4-e5f6-7890-abcd-ef1234567890"
)
asslist <- assignmentClass(ass_list)

# retrieve the uid of the person responsible
getinfo(asslist, "responsibleid")

# see all available attribute names
getinfo(asslist, "arglist")
```

is.assignmentClass	<i>Function to check if an object is of class assignmentClass</i>
--------------------	---

Description

Function to check if an object is of class assignmentClass

Usage

```
is.assignmentClass(x)
```

Arguments

x	object to be checked
---	----------------------

Value

TRUE if object is of class assignmentClass

Examples

```
ac <- assignmentClass(list(Assignments = data.frame(Id = 1)))
is.assignmentClass(ac)
is.assignmentClass(data.frame())
```

is.exportClass	<i>Function to check if an object is of class exportClass</i>
----------------	---

Description

Function to check if an object is of class exportClass

Usage

```
is.exportClass(x)
```

Arguments

x	object to be checked
---	----------------------

Value

TRUE if object is of class exportClass

Examples

```
dt <- data.table::data.table(q1 = 1:5)
vlabs <- data.table::data.table(VariableName = "q1", QuestionText = "Question 1")
ec <- exportClass(dt, varLabels = vlabs)
is.exportClass(ec)
is.exportClass(dt)
```

is.UserClass	<i>Function to check if an object is of class UserClass</i>
--------------	---

Description

Function to check if an object is of class UserClass

Usage

```
is.UserClass(x)
```

Arguments

x	object to be checked
---	----------------------

Value

TRUE if object is of class UserClass

Examples

```
uc <- UserClass(list(Users = data.frame(UserName = "interviewer1")))
is.UserClass(uc)
is.UserClass(data.frame())
```

summaryTable	<i>S3 method to create a summary table of numeric variables for an exportClass object</i>
--------------	---

Description

This function generates a summary table of numeric variables in an exportClass object.

Usage

```
summaryTable(x, ...)

## S3 method for class 'exportClass'
summaryTable(x, ..., includeFactors = FALSE, useDT = TRUE, DTstyle = TRUE)

## S3 method for class 'paradata'
summaryTable(x, ..., useDT = TRUE, DTstyle = TRUE)
```

Arguments

x	exportClass object.
...	further arguments.
includeFactors	if TRUE, factor variables will be converted to numeric and included in the table.
useDT	Logical, if TRUE (default) the function returns a DataTable (DT) object (HTML table), otherwise a data.table object is returned.
DTstyle	if TRUE and useDT = TRUE, then a custom style will be applied, otherwise a plain DT table will be returned.

Details

- For the main data: The table includes statistics such as mean, standard deviation, maximum, minimum, and total count as well as count of NA values.
- For paradata: Different summary tables for response times, duration times etc. see examples.

Value

A DataTable (DT) object (HTML table) displaying the summary statistics, if useDT = TRUE, and a plain data.table otherwise.

Examples

```

questlist<-suso_getQuestDetails()

### MAIN DATA
export <- suso_export(questID = questlist$QuestionnaireId[7],
                      version = questlist$Version[7],
                      combineFiles = TRUE,
                      reloadTimeDiff = 20)

# Create a summary table
summary_data_table <- summaryTable(export)

# Display the summary table
summary_data_table

dt <- data.table::data.table(q1 = 1:10, q2 = 11:20)
vlabs <- data.table::data.table(
  VariableName = c("q1", "q2"),
  QuestionText = c("Question 1", "Question 2")
)
ec <- exportClass(dt, varLabels = vlabs)
summaryTable(ec, useDT = FALSE)

questlist<-suso_getQuestDetails()

### PARADATA
# 1. Pre-Processed tables created during export
para<-suso_export_paradata(questID = questlist$QuestionnaireId[1],
                          version = questlist$Version[1], reloadTimeDiff = 24,
                          workStatus = "All", asList = FALSE, onlyActiveEvents = TRUE)

# Create a summary table of all event counts by action (includes always passive and active one)
actions <- summaryTable(para, "action")

# Create a summary table of all event counts by user
users <- summaryTable(para, "user")

# 2. Tables created on request for response times
# grouped by variables, i.e. type, VariableName, responsible etc.

# Create a summary table of the average response time by variable
resptime_var <- summaryTable(para, "VariableName")

# Create a summary table of the average response time by question type
resptime_type <- summaryTable(para, "type")

# Create a summary table of the average response time by user
resptime_user <- summaryTable(para, "responsible")

```

summaryTable.UserClass

S3 method to create a summary table of users for UserClass object

Description

This function generates a summary table of Users.

Usage

```
## S3 method for class 'UserClass'
summaryTable(x, includeFactors = FALSE, useDT = TRUE, DTstyle = TRUE, ...)
```

Arguments

x	A UserClass object.
includeFactors	if TRUE, factor variables will be converted to numeric and included in the table (only for exportClass).
useDT	Logical, if TRUE (default) the function returns a DataTable (DT) object (HTML table), otherwise a data.table object is returned.
DTstyle	if TRUE and useDT = TRUE, then a custom style will be applied, otherwise a plain DT table will be returned.
...	Additional arguments (not used).

Details

For very large tables you will get a warning when rendering in Rstudio, however in a shiny app you have the option to use server side processing: `DT::renderDataTable(..., server = TRUE)`. In cases where `future` and `future.apply` is available on your system the additional information will be retrieved in parallel (multisession), otherwise sequential. For large data sets parallel processing is recommended. Furthermore when processing in parallel, a progress bar for shiny applications is included and active when running in a shiny application, otherwise a cli progress bar will be displayed.

Value

A DataTable (DT) object (HTML table) if `useDT = TRUE`, and a plain `data.table` otherwise. The table contains all users core data, plus completed and assigned counts,

Examples

```
allintprim<-sus.o_getINT(workspace = "primary")

# Create a summary table
summary_data_table <- summaryTable(allintprim)

# Display the summary table
summary_data_table
```

sus.o.options

Configuration options used for the SurveySolutionsAPIv2 package

Description

Below are the options and environment variables, that are used by the **SurveySolutionsAPIv2** package.

Value

No return value, called for documentation on package configuration options.

Available options

Bellow is a list of options, with their default values.

- ‘sus.o.url.http’ if TRUE allows for non SSL connections, which is generally not recommended for publicly facing set-ups, however may be required for testing purposes.
- ‘sus.o.maxpar.req’ specifies the maximum number of parallel requests, default is 100.
- ‘sus.o.maxpar.con’ specifies the maximum number of parallel connections, and must always be equal or greater than sus.o.maxpar.req, default is 100.
- ‘sus.o.para.break’ specifies the maximum response time to be considered as a break in seconds, default 120.
- ‘sus.o.para.tz’ specifies the local timezone for the processing of time values, default is Sys.timezone().
- ‘sus.o.para.maxcore’ specifies the number of cores used for parallel processing, default is data.table::getDTthreads()-2.
- ‘sus.o.para.plan’ specifies plane used for parallel processing, default is multisession. For details please see future::plan
- ‘sus.o.uses shiny’ should R shiny elements be used if running in shiny app, default is TRUE. Currently implemented for:
 - [sus.o_PwCheck](#), providing feedback for credentials check, the ‘sus.o.pwcheck.message_succ’ and ‘sus.o.pwcheck.message_fail’ can be modified.

- `suso_export`, providing different progress bars for interactive (cli) use and in shiny app use (shiny::`withProgress`, to also customize the progress bar message, i.e. in a different language, the `'suso.progressbar.message'` can be modified.
- TBD

Examples

```
# Inspect the current maximum parallel requests option
getOption("suso.maxpar.req", default = 100)

# Inspect the timezone option
getOption("suso.para.tz", default = Sys.timezone())
```

suso_archUSR	<i>Survey Solutions API call (un-) archive user</i>
--------------	---

Description

(Un-)Archive user

Usage

```
suso_archUSR(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  user_id = NULL,
  archive = FALSE,
  workspace = suso_get_api_key("workspace"),
  token = NULL
)
```

Arguments

<code>server</code>	Survey Solutions server address
<code>apiUser</code>	Survey Solutions API user
<code>apiPass</code>	Survey Solutions API password
<code>user_id</code>	user id
<code>archive</code>	if TRUE user will be archived or statys archived, if FALSE user will be unarchived or stays unarchived
<code>workspace</code>	If workspace name is provide requests are made regarding this specific workspace, if no workspace is provided defaults to primary workspace.
<code>token</code>	If Survey Solutions server token is provided <i>usr</i> and <i>pass</i> will be ignored

Value

A data.table containing the user ID, archive timestamp, and archive status.

Examples

```
# you can archive a user by archive=T
susos_archUSR(
  workspace = "myworkspace",
  user_id = "xxxx-xxxx-xxxx-xxx",
  archive = TRUE
)
# and unarchive a user by archive=F
susos_archUSR(
  workspace = "myworkspace",
  user_id = "xxxx-xxxx-xxxx-xxx",
  archive = FALSE
)
```

`susos_assignInterview` *Assign interview to interviewer or supervisor*

Description

Reassigns an interview to a specified interviewer or supervisor.

Usage

```
susos_assignInterview(
  server = susos_get_api_key("susosServer"),
  apiUser = susos_get_api_key("susosUser"),
  apiPass = susos_get_api_key("susosPass"),
  workspace = susos_get_api_key("workspace"),
  token = NULL,
  intID = "",
  userId = NULL,
  userName = NULL,
  supervisor = FALSE
)
```

Arguments

<code>server</code>	Survey Solutions server address
<code>apiUser</code>	Survey Solutions API user
<code>apiPass</code>	Survey Solutions API password

workspace	server workspace name
token	API token
intID	InterviewId (GUID)
userId	User GUID of responsible person
userName	Name of responsible person (optional)
supervisor	If TRUE assigns to a supervisor, otherwise assigns to an interviewer

Value

A data.table indicating the assignment status.

Examples

```
suso_assignInterview(
  intID = "11111111-2222-3333-4444-555555555555",
  userName = "interviewer01"
)
```

suso_assignWorkspace *Survey Solutions API call to assign workspace (!! WORKS ONLY WITH ADMIN CREDENTIALS)*

Description

suso_assignWorkspace Allows you to assign a workspace to a specific user or a group of users. For more details please read <https://docs.mysurvey.solutions/headquarters/accounts/workspaces/>. To run this command you require admin credentials and not the regular API user credentials.

Usage

```
suso_assignWorkspace(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  token = NULL,
  assign_workspace = NULL,
  keep_old_workspace = TRUE,
  uid = NULL,
  sv_id = NULL
)
```

Arguments

<code>server</code>	Survey Solutions server address.
<code>apiUser</code>	Survey Solutions ADMIN user.
<code>apiPass</code>	Survey Solutions ADMIN password.
<code>token</code>	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored.
<code>assign_workspace</code>	The workspace which you want to assign to the new user.
<code>keep_old_workspace</code>	if TRUE, existing assigned workspaces will be kept, if FALSE only the new one will be assigned, see details.
<code>uid</code>	Either a single User ID of the user to be assigned, or a vector.
<code>sv_id</code>	The supervisor's ID to which the interviewer should be assigned to, if it is a supervisor who is assigned, just use the same as in <i>uid</i> . Must be the same length as <i>uid</i>

Details

When `keep_old_workspace=FALSE` the user will only be assigned to the new workspace, otherwise the existing ones will be added, if the former is the case, then you need to make sure, that all assignments, interviews, team members etc. are cleared in the old workspace. Be aware, that for using this call you require admin credentials, and not the regular API user credentials.

Value

If successful, returns a data.table with the details as well as the Status message workspace list updated.

A data.table containing the assignment update status.

Examples

```
## Use Admin Credentials!

# Assign a new workspace to a single user, keep old one(s)
suso_assignWorkspace(
  assign_workspace = "myworkspace1",
  uid = "xxx-xxx-xxx-xxx-xxx",
  sv_id = "xxx-xxx-xxx-xxx-xxx",
  apiUser = "xxxxxx",
  apiPass = "xxxxxx",
  keep_old_workspace = TRUE
)

# Assign a new workspace to a single user, drop old one(s)
suso_assignWorkspace(
  assign_workspace = "myworkspace1",
  uid = "xxx-xxx-xxx-xxx-xxx",
```

```

        sv_id = "xxx-xxx-xxx-xxx-xxx",
        apiUser = "xxxxxx",
        apiPass = "xxxxxx",
        keep_old_workspace = TRUE
    )

# Assign all supervisors from on workspace to a new one, keep old one
allsv<-suso_getSV(workspace = "old")
suso_assignWorkspace(
    assign_workspace = "new",
    uid = allsv$UserId,
    sv_id = allsv$sv_id,
    apiUser = "xxxxxx",
    apiPass = "xxxxxx",
    keep_old_workspace = TRUE
)

# Assign all interviewers from one workspace to a new one, keep old one
allint<-suso_getINT(workspace = "old")
suso_assignWorkspace(
    assign_workspace = "new",
    uid = allint$UserId,
    sv_id = allint$sv_id,
    apiUser = "xxxxxx",
    apiPass = "xxxxxx",
    keep_old_workspace = TRUE
)

```

suso_cancelExport	<i>Survey Solutions API call to cancel an export process</i>
-------------------	--

Description

Cancels an ongoing export process or deletes an export process by ID.

Usage

```

suso_cancelExport(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  token = NULL,
  workspace = suso_get_api_key("workspace"),
  jobid = NULL
)

```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
workspace	server workspace, if nothing provided, defaults to primary
jobid	Export process ID (integer)

Value

A data.table containing the canceled export process information.

Examples

```
# Cancel an ongoing export process
suso_cancelExport(jobid = 123)
```

suso_clear_keys	<i>Clear Credentials</i>
-----------------	--------------------------

Description

Clears all the API credentials

Usage

```
suso_clear_keys()
```

Value

Invisible NULL.

Examples

```
suso_clear_keys()
```

susorcommentInterview *Leave a comment on a question in an interview*

Description

Leaves a comment on a question using either the questionId or the questionnaire variable name.

Usage

```
susorcommentInterview(  
  server = suso_get_api_key("susoserver"),  
  apiUser = suso_get_api_key("susouser"),  
  apiPass = suso_get_api_key("susopass"),  
  workspace = suso_get_api_key("workspace"),  
  token = NULL,  
  intID = "",  
  comment = "",  
  questionId = NULL,  
  variable = NULL,  
  rosterVector = NULL  
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace name
token	API token
intID	InterviewId (GUID)
comment	The comment text to leave
questionId	The question GUID (optional if variable is provided)
variable	Variable name of question (optional if questionId is provided)
rosterVector	Integer vector specifying roster indices (e.g. c(0, 1))

Value

A data.table indicating the comment status.

Examples

```
suso_commentInterview(
  intID = "11111111-2222-3333-4444-555555555555",
  variable = "age",
  comment = "Please verify respondent age."
)
```

suso_createASS

Survey Solutions API call for Assignment Creation

Description

Creates assignment with ID data. Uses the `httr2` package, and specifically the [req_perform_parallel](#) function. Compared to a sequential approach, this significantly and safely decreases the overall processing time. Adjust relevant option `'suso.maxpar.req'` for number of parallel processes to match the capacity of your system (default is 100 parallel requests).

Usage

```
suso_createASS(
  df = NULL,
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  workspace = suso_get_api_key("workspace"),
  token = NULL,
  questID = NULL,
  version = NULL
)
```

Arguments

<code>df</code>	dataframe with upload data for ID (identifying data) assignments, see details for structure.
<code>server</code>	Survey Solutions server address.
<code>apiUser</code>	Survey Solutions API user.
<code>apiPass</code>	Survey Solutions API password.
<code>workspace</code>	server workspace, if nothing provided, defaults to primary.
<code>token</code>	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored.
<code>questID</code>	the questionnaire id.
<code>version</code>	the questionnaire version.

Details

Dataframe needs to be provided with columns for ID data, matching the required type, as well as *Quantity* and *ResponsibleName*. Return value is a data.table, with the ID data, if successful.

Value

Returns a data.table with a row for each assignment, containing identifying data, responsible id etc.

Examples

```
# get the list of questionnaires in the workspace
questlist<-susos_getQuestDetails()

# Create the assignments with your upload data
asslist <- susos_createASS(df = IdentifyingData,
                          questID = questlist$QuestionnaireId[1],
                          version = questlist$Version[1])
```

`susos_createUSER`*Survey Solutions API call for User Creation*

Description

Creates Survey Solutions users (observers, interviewers or supervisors).

Usage

```
susos_createUSER(
  userlist = NULL,
  server = susos_get_api_key("susosServer"),
  apiUser = susos_get_api_key("susosUser"),
  apiPass = susos_get_api_key("susosPass"),
  workspace = susos_get_api_key("workspace"),
  token = NULL,
  showUser = FALSE
)
```

Arguments

<code>userlist</code>	dataframe with upload data
<code>server</code>	Survey Solutions server address
<code>apiUser</code>	Survey Solutions API user
<code>apiPass</code>	Survey Solutions API password
<code>workspace</code>	server workspace, if nothing provided, defaults to primary
<code>token</code>	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
<code>showUser</code>	userName in console

Details

Dataframe needs to be provided with the mandatory columns for user creation which are: Role, UserName, Password and Supervisor (in case of interviewer), optional you can also provide Full-Name, PhoneNumber and Email. Return value is a data.table, which includes the user information as well as the response's status code. Important is also that the UserName and Password are provided in the required format.

Value

A data.table containing the user information and HTTP response status code for each created user.

Examples

```
new_users <- data.frame(
  Role = "Interviewer",
  UserName = "interviewer01",
  Password = "Password123!",
  Supervisor = "supervisor01"
)
suso_createUSER(userlist = new_users)
```

`suso_createWorkspace` *Survey Solutions API call to create workspace*

Description

`suso_createWorkspace` Allows you to create a workspace with a specific system name and display name. For more details please read <https://docs.mysurvey.solutions/headquarters/accounts/workspaces/>. To run this command you require admin credentials.

Usage

```
suso_createWorkspace(  
  server = suso_get_api_key("susoServer"),  
  apiUser = suso_get_api_key("susoUser"),  
  apiPass = suso_get_api_key("susoPass"),  
  token = NULL,  
  new_workspace = NULL,  
  displayName = NULL  
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions ADMIN user
apiPass	Survey Solutions ADMIN password
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
new_workspace	The name used by the system for this workspace. Make sure you follow the naming rules outlined under https://docs.mysurvey.solutions/headquarters/accounts/workspaces/
displayName	The name visible to the users

Details

Be aware, that for using this call you require the ADMIN credentials, and not the regular API user credentials.

Value

A data.table containing the created workspace information.

Examples

```
# Use Admin Credentials!  
suso_createWorkspace(  
  new_workspace = "myworkspace1",  
  displayName = "SpecialWorkspace",  
  apiUser = "xxxxxx",  
  apiPass = "xxxxxx")
```

suso_deleteInterview	<i>Delete an interview</i>
----------------------	----------------------------

Description

Deletes an interview from the Survey Solutions server.

Usage

```
suso_deleteInterview(  
  server = suso_get_api_key("susoServer"),  
  apiUser = suso_get_api_key("susoUser"),  
  apiPass = suso_get_api_key("susoPass"),  
  workspace = suso_get_api_key("workspace"),  
  token = NULL,  
  intID = ""  
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace name
token	API token
intID	InterviewId of the interview (GUID)

Value

A data.table indicating the deletion status.

Examples

```
suso_deleteInterview(intID = "11111111-2222-3333-4444-555555555555")
```

suso_deletemap	<i>Delete Map from Server</i>
----------------	-------------------------------

Description

Allows the user to delete maps stored on the server.

Usage

```
suso_deletemap(  
  server = suso_get_api_key("susoServer"),  
  apiUser = suso_get_api_key("susoUser"),  
  apiPass = suso_get_api_key("susoPass"),  
  workspace = suso_get_api_key("workspace"),  
  token = NULL,  
  fileName = NULL  
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
fileName	the name of the map file on the server

Value

A data.table containing the deleted map details (file name, shape type, and import timestamp).

Examples

```
suso_deletemap(  
  workspace = "myworkspace",  
  fileName = "Lat9264Lon625_ALL.tif"  
)
```

`susu_deleteWorkspace` *Delete a workspace*

Description

Deletes a specified workspace. Accessible only to administrator.

Usage

```
susu_deleteWorkspace(
  server = susu_get_api_key("susuServer"),
  apiUser = susu_get_api_key("susuUser"),
  apiPass = susu_get_api_key("susuPass"),
  token = NULL,
  workspace = NULL
)
```

Arguments

<code>server</code>	Survey Solutions server address
<code>apiUser</code>	Survey Solutions API user
<code>apiPass</code>	Survey Solutions API password
<code>token</code>	API token
<code>workspace</code>	Name of the workspace to delete

Value

A data.table indicating the deletion status.

Examples

```
susu_deleteWorkspace(workspace = "old_ws")
```

`susu_enableWorkspace` *Enable or disable a workspace*

Description

Enables or disables access to a specified workspace. Accessible only to administrator.

Usage

```
suso_enableWorkspace(  
  server = suso_get_api_key("susoServer"),  
  apiUser = suso_get_api_key("susoUser"),  
  apiPass = suso_get_api_key("susoPass"),  
  token = NULL,  
  workspace = NULL,  
  enable = TRUE  
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
token	API token
workspace	Name of the workspace
enable	Logical, if TRUE enables the workspace, if FALSE disables it. Default is TRUE.

Value

A data.table indicating the enable/disable status.

Examples

```
# Enable workspace  
suso_enableWorkspace(workspace = "my_ws", enable = TRUE)  
# Disable workspace  
suso_enableWorkspace(workspace = "my_ws", enable = FALSE)
```

suso_export	<i>Survey Solutions API call to generate and download the data</i>
-------------	--

Description

Generates and downloads the data from your Survey Solutions server.

Usage

```

suso_export(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  token = NULL,
  workspace = suso_get_api_key("workspace"),
  questID = NULL,
  version = NULL,
  from_date = NULL,
  from_time = "00:00:00",
  to_date = NULL,
  to_time = "23:59:59",
  workStatus = c("All", "SupervisorAssigned", "InterviewerAssigned",
    "RejectedBySupervisor", "Completed", "ApprovedBySupervisor",
    "RejectedByHeadquarters", "ApprovedByHeadquarters"),
  addTranslation = FALSE,
  translationLanguage = NULL,
  reloadTimeDiff = 1,
  inShinyApp = FALSE,
  verbose = FALSE,
  weight_file = NULL,
  process_mapquestions = FALSE,
  combineFiles = TRUE,
  addsysfiles = FALSE
)

```

Arguments

<code>server</code>	Survey Solutions server address
<code>apiUser</code>	Survey Solutions API user
<code>apiPass</code>	Survey Solutions API password
<code>token</code>	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
<code>workspace</code>	server workspace, if nothing provided, defaults to primary
<code>questID</code>	Questionnaire ID
<code>version</code>	Questionnaire version
<code>from_date</code>	if provided, only interviews started on this date or later will be included
<code>from_time</code>	if provided, only interviews started at this time or later will be included
<code>to_date</code>	if provided, only interviews started before or on this date will be included
<code>to_time</code>	if provided, only interviews started before or at this time will be included
<code>workStatus</code>	define which statuses the file should include (i.e. <i>Restored, Created, SupervisorAssigned, InterviewerAssigned, RejectedBySupervisor, ReadyForInterview, SentToCapi, Restarted, Completed, ApprovedBySupervisor, RejectedByHeadquarters, ApprovedByHeadquarters, Deleted</i>), if NULL all is exported

<code>addTranslation</code>	if not NULL, the translation name as specified in the designer, which will then be applied to value and variable labels
<code>translationLanguage</code>	if not NULL the desired translation to be applied, if NULL, the first one found will be applied, the same is true, if the provided translation language does not exist
<code>reloadTimeDiff</code>	time difference in hours between last generated file and now (will be ignored when <code>from_date</code> and <code>to_date</code> is not NULL)
<code>inShinyApp</code>	if TRUE, file interacts with shiny progress bar. <i>DEPRECATED, not used any longer, only for compatability reasons with previous SurveySolutionsAPI R package.</i>
<code>verbose</code>	if TRUE, prints out information about the progress
<code>weight_file</code>	file path to file with survey weights. if provided, the weights will be added to the export
<code>process_mapquestions</code>	(only when <code>combineFiles=FALSE</code>), should map questions be processed to spatial (sf) objects, if yes, gps and all types of mapquestions will be added to the list at their corresponding roster level, and with the prefix "sf_" (<i>experimental!</i>)
<code>combineFiles</code>	if TRUE, the export will be combined into single data.table, see details for the processing steps
<code>addsysfiles</code>	if TRUE Survey Solutions system files, i.e. • <code>assignment__actions.tab</code> • <code>interview__actions.tab</code> • <code>interview__comments.tab</code> • <code>interview__diagnostics.tab</code> • <code>interview__errors.tab</code> are also included. Ignored when <code>combineFiles=TRUE</code> .

Details

This API call uses the tab export format and uses information from the included questionnaire document to assign value labels to any factor variables. Currently this is done for single select and multiselect questions. If you export the data with `combineFiles = FALSE` a list will be returned with the following structure:

- it is returned as a LIST with up to 4 different lists. The list names are:
 - *main* Contains the top level data, and (if available interviewer comments)
 - *R1* All rosters in roster level 1
 - *R2* All rosters in roster level 2
 - *R3* All rosters in roster level 3
- Number of lists depends on the level of roster nesting
- If `process_mapquestions = TRUE` is provided, the different lists will also contain an sf object for each spatial variable found at the corresponding main/roster level (multipolygons, multi-points, points and lines), for details see also the [sf](#) package.

- All variable names are transformed to lower case and categorical variables are consistently labeled
- List elements are returned as data.tables
- Allows for specification of reload time (i.e. generation of new download file)
- PRESERVES categorical labels *and* values.

If however, you export the data with *combineFiles = TRUE* a single data.table containing all the data in wide format will be returned. This is the result of the following processing steps:

- First all roster files are cast into wide format.
- Second the rosters are merged from bottom to top.
- Third the result of the previous step is merged with the main file

The resulting data.table contains value labels for factor variables. If the file path to a weight file is provided, then these will be added too, and is as such ready for analysis.

Value

a single exportClass data.table or a list of exportClass data.tables, see details.

Examples

```
questlist<-susso_getQuestDetails()

exp<-susso_export(questID = questlist$QuestionnaireId[1],
  version = questlist$Version[1],
  workStatus = "All", process_mapquestions = TRUE,
  combineFiles = TRUE,
  reloadTimeDiff = 0,
  translationLanguage = "italian")
```

susso_export_paradata *Export Survey Solutions paradata*

Description

Exports Survey Solutions paradata, and returns either a single exportClass data.table, or a list of data.tables.

Usage

```

suso_export_paradata(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  token = NULL,
  workspace = suso_get_api_key("workspace"),
  questID = NULL,
  version = NULL,
  from_date = NULL,
  from_time = "00:00:00",
  to_date = NULL,
  to_time = "23:59:59",
  workStatus = c("All", "SupervisorAssigned", "InterviewerAssigned",
    "RejectedBySupervisor", "Completed", "ApprovedBySupervisor",
    "RejectedByHeadquarters", "ApprovedByHeadquarters"),
  reloadTimeDiff = 1,
  inShinyApp = FALSE,
  multiCore = NULL,
  onlyActiveEvents = FALSE,
  asList = FALSE,
  allResponses = TRUE,
  gpsVarName = NA,
  verbose = FALSE,
  showProgress = FALSE
)

```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
workspace	server workspace, if nothing provided, defaults to primary
questID	<i>QuestionnaireId</i> for which the paradata should be generated
version	questionnaire version
from_date	if provided, only interviews started on this date or later will be included
from_time	if provided, only interviews started at this time or later will be included
to_date	if provided, only interviews started before or on this date will be included
to_time	if provided, only interviews started before or at this time will be included
workStatus	define which statuses the file should include (i.e. <i>Restored, Created, SupervisorAssigned, InterviewerAssigned, RejectedBySupervisor, ReadyForInterview, SentToCapi, Restarted, Completed, ApprovedBySupervisor, RejectedByHeadquarters, ApprovedByHeadquarters, Deleted</i>), if NULL all is exported

<code>reloadTimeDiff</code>	time difference in hours between last generated file and now (will be ignored when <code>from_date</code> and <code>to_date</code> is not NULL)
<code>inShinyApp</code>	if TRUE, file interacts with shiny progress bar
<code>multiCore</code>	if not NULL, an integer number specifying the number of cores to use
<code>onlyActiveEvents</code>	if TRUE only active events are exported, decreases processing time and memory requirements
<code>asList</code>	only used if <code>onlyActiveEvents = T</code> , if TRUE returns a list with a separate list element for each event, if FALSE returns a single <i>exportClass</i> data.table object. See details for more information.
<code>allResponses</code>	if TRUE all responses will be extracted. Setting it to FALSE may decrease processing time and memory requirements
<code>gpsVarName</code>	provide GPS variable name. If not provided, identification is attempted by lookin for a variable containing gps in its name.
<code>verbose</code>	if TRUE, shows messages about the operation carried out. Can be useful for longrunning operations.
<code>showProgress</code>	also display the progress bars.

Details

`suso_export_paradata` returns a data.table. Calculates the response time and separates multiple responses into individual columns. It also creates a variable *counter* which preserves the sequence of events.

The return value is a list with a separate list element for each event. If any of the variable names contains *gps* this function also attempts to identify (and extract) the geo-reference location. In case of multiple *gps* variables, it identifies the first variable, with not all missing values. This in turn facilitates the creation of paradata maps (see vignette on paradata). In addition it also returns all the variables and responses in separate columns and as factors. Exporting *onlyActiveEvents* substantially decrease processing time. The events may be sufficient for most of the paradata analysis.

To further decrease the processing time, one could set *allResponses* to FALSE. Doing so will still export all the data, however it will not attempt to extract all responses and setting them to factors.

Value

a single *exportClass* data.table or a list of data.tables.

Examples

```
questlist<-suso_getQuestDetails()
# Get a single data.table with response timings,
# only active events,
# and responses are not expanded
para<-suso_export_paradata(questID = questlist$QuestionnaireId[1],
                           version = questlist$Version[1],
                           reloadTimeDiff = 0,
                           workStatus = "All",
```

```
asList = FALSE,
onlyActiveEvents = TRUE,
allResponses = FALSE)

# Create a summary table
summary_data_table <- summaryTable(para)

# Display the summary table
summary_data_table
```

suso_get_api_key	<i>Get credentials</i>
------------------	------------------------

Description

Get API credentials

Usage

```
suso_get_api_key(api = c("susoServer", "susoUser", "susoPass", "workspace"))
```

Arguments

api one of susoServer, susoUser, susoPass, or workspace

Details

Get credentials, used as input in API calls

Value

Character string of the requested credential, or NA if not set.

Examples

```
suso_get_api_key("susoServer")
suso_get_api_key("workspace")
```

suso_get_assignments *Survey Solutions API call for assignment list*

Description

suso_get_assignments calls the Survey Solutions assignment API

Usage

```
suso_get_assignments(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  token = NULL,
  workspace = suso_get_api_key("workspace"),
  questID = NULL,
  AssId = NULL,
  version = NULL,
  responsibleID = NULL,
  supervisorId = NULL,
  searchBy = NULL,
  status = NULL,
  start = NULL,
  length = NULL,
  ShowArchive = FALSE,
  order.by = c("Id ASC", "Id DESC", "ResponsibleName DESC", "ResponsibleName ASC",
    "InterviewsCount DESC", "InterviewsCount ASC", "Quantity DESC", "Quantity ASC",
    "UpdatedAtUtc DESC", "UpdatedAtUtc ASC", "CreatedAtUtc DESC", "CreatedAtUtc ASC"),
  operations.type = c("assignmentQuantitySettings", "history", "recordAudio")
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
workspace	If workspace name is provide requests are made regarding this specific workspace
questID	only assignments for <i>QuestionnaireId</i> are returned, requires version being not NULL
AssId	if NULL a list of all assignments on the server, if not NULL the assignment details for a specific assignment ID
version	version of the questionnaire, only required with questID
responsibleID	the ID of the responsible user (Supervisor or Interviewer). Retrieves all assignments for this user.

<code>supervisorId</code>	Filter assignments by supervisor ID
<code>searchBy</code>	Filter result by custom search query
<code>status</code>	Filter assignments by status (e.g. NotAssigned, Assigned, Closed, Archived, Deleted, Completed)
<code>start</code>	start index for history query (only with <code>operations.type = "history"</code>)
<code>length</code>	page length for history query (only with <code>operations.type = "history"</code>)
<code>ShowArchive</code>	if TRUE, only archived assignments are included in the list
<code>order.by</code>	determines the column by which the assignment list should be ordered, one of <i>Id</i> , <i>ResponsibleName</i> , <i>InterviewsCount</i> , <i>Quantity</i> , <i>UpdatedAtUtc</i> , <i>CreatedAtUtc</i> , followed by ordering direction "ASC" or "DESC", e.g. "Id DESC" or "CreatedAtUtc ASC"
<code>operations.type</code>	specifies the desired operation, one of <code>assignmentQuantitySettings</code> , <code>history</code> , or <code>recordAudio</code> , if specified, requires also <i>AssId</i> to be specified.

Value

Returns an S3 object of `assignmentClass`. If you select any of the operations types, then no `data.frame` is returned, the `data.table` will be NULL, however any information returned from the API can be retrieved by using the `getinfo()` function with the corresponding arguments.

Examples

```
# get all assignment for specific workspace
asslist<-suso_get_assignments(
  workspace = "myworkspace"
)

# get all assignment for specific responsible
asslist<-suso_get_assignments(
  workspace = "myworkspace",
  responsibleID = "a67d2b82-bf28-40cf-bd1a-7901225c0885"
)

# get the overall count
getinfo(asslist, "totalcount")

# get all single assignment details
asslist<-suso_get_assignments(
  workspace = "myworkspace",
  AssId = 1
)

# retrieve the uid of the person responsible
getinfo(asslist, "responsibleid")

# get the status of the quantitysettings
asslist<-suso_get_assignments(
  workspace = "myworkspace",
  AssId = 1,
```

```
      operations.type = "assignmentQuantitySettings")
asslist
# Null data.table (0 rows and 0 cols)

getinfo(asslist, "canchangequantity")
# [1] TRUE
```

susoset_key *Checks if credentials are present*

Description

Helper function

Usage

```
susoset_key(
  api = c("susoserver", "susouser", "susokey", "workspace")
)
```

Arguments

api one of susoserver, susouser, susokey, or workspace

Value

Character string of the requested credential, or throws error if not set.

Examples

```
susoset_key(
  susoserver = "https://demo.mysurvey.solutions",
  susouser = "api_user",
  susopassword = "password123",
  workspace = "primary"
)
susoset_key("susoserver")
susoset_keys()
```

suso_get_stats

Survey Solutions API call for Summary Tables

Description

Returns summary tables for individual questions from Survey Solutions statistics. If no responses have been provided, an empty table will be returned.

Usage

```
suso_get_stats(  
  server = suso_get_api_key("susoServer"),  
  apiUser = suso_get_api_key("susoUser"),  
  apiPass = suso_get_api_key("susoPass"),  
  workspace = suso_get_api_key("workspace"),  
  token = NULL,  
  questID = NULL,  
  version = NULL,  
  qQuest = "",  
  byTeam = TRUE,  
  exportType = c("Csv", "Json")  
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
questID	Questionnaire ID (GUID)
version	Version of the questionnaire (numeric)
qQuest	Question variable name or UUID
byTeam	Logical, whether the table should contain reports by team (default: TRUE)
exportType	Report format: "Csv" (default) or "Json"

Value

A data.table containing the summary statistics for the question.

Examples

```
suso_get_stats(  
  questID = "11111111-2222-3333-4444-555555555555",  
  version = 1,  
  qQuest = "age"  
)
```

suso_get_stats_interview
<i>Get statistics for interview</i>

Description

Fetch statistics for a specific interview, vectorized over interview id (int_id)

Usage

```
suso_get_stats_interview(  
  server = suso_get_api_key("susoServer"),  
  apiUser = suso_get_api_key("susoUser"),  
  apiPass = suso_get_api_key("susoPass"),  
  workspace = suso_get_api_key("workspace"),  
  token = NULL,  
  intID = ""  
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
intID	a single or multiple <i>InterviewId</i> .

Value

A data.table containing statistics for the requested interview(s).

Examples

```
suso_get_stats_interview(  
    workspace = "myworkspace",  
    intID = "dee7705f-d611-4b12-9b97-2b8e5b80c4ea"  
)
```

suso_getAllAnswerInterview	<i>Survey Solutions API call to retrieve all answers for a specific interview</i>
----------------------------	---

Description

Returns all responses for a specific interview

Usage

```
suso_getAllAnswerInterview(  
    server = suso_get_api_key("susoServer"),  
    apiUser = suso_get_api_key("susoUser"),  
    apiPass = suso_get_api_key("susoPass"),  
    workspace = suso_get_api_key("workspace"),  
    token = NULL,  
    intID = ""  
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
intID	the <i>InterviewId</i> of the interview. To get a list of all interview for a specific questionnaire, execute <i>suso_getAllInterviewQuestionnaire</i>

Value

A data.table containing all answers for the given interview.

Examples

```
suso_getAllAnswerInterview(
  workspace = "myworkspace",
  intID = "dee7705f-d611-4b12-9b97-2b8e5b80c4ea"
)
```

```
suso_getAllHistoryInterview
```

Get all history for a specific interview

Description

Get all history for a specific interview

Usage

```
suso_getAllHistoryInterview(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  workspace = suso_get_api_key("workspace"),
  token = NULL,
  intID = ""
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
intID	the <i>InterviewId</i> of the interview.

Value

A data.table containing the history records for the interview.

Examples

```
suso_getAllHistoryInterview(  
    workspace = "myworkspace",  
    intID = "dee7705f-d611-4b12-9b97-2b8e5b80c4ea"  
)
```

suso_getAllInterviewQuestionnaire	<i>Survey Solutions API call to retrieve all interviews for a specific questionnaire</i>
-----------------------------------	--

Description

Returns all interviews for the specified questionnaire and the selected status.

Usage

```
suso_getAllInterviewQuestionnaire(  
    server = suso_get_api_key("susoServer"),  
    apiUser = suso_get_api_key("susoUser"),  
    apiPass = suso_get_api_key("susoPass"),  
    workspace = suso_get_api_key("workspace"),  
    token = NULL,  
    questID = "",  
    version = 1,  
    workStatus = c("Completed", "All", "SupervisorAssigned", "InterviewerAssigned",  
        "RejectedBySupervisor", "ApprovedBySupervisor", "RejectedByHeadquarters",  
        "ApprovedByHeadquarters")  
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
questID	your Survey Solutions <i>QuestionnaireId</i> . Retrieve a list of questionnaires by executing <i>suso_getQuestDetails</i>
version	version of the questionnaire
workStatus	define which statuses the file should include (i.e. <i>Restored, Created, SupervisorAssigned, InterviewerAssigned, RejectedBySupervisor, ReadyForInterview, SentToCapi, Restarted, Completed, ApprovedBySupervisor, RejectedByHeadquarters, ApprovedByHeadquarters, Deleted</i>), if NULL only completed interviews will be shown.

Details

ATTENTION: This function only exists for consistency reasons with the original SurveySolution-SAPI package. Under the hood it uses [suso_getQuestDetails](#), which is based on the GraphQL API.

Value

A data.table listing all interviews matching the specified questionnaire and status filter.

Examples

```
# To get all interviews for a specific questionnaire and a specific status
suso_getAllInterviewQuestionnaire(
  workspace = "myworkspace",
  questID = "dee7705f-d611-4b12-9b97-2b8e5b80c4ea",
  version = 1,
  workStatus = "InterviewerAssigned"
)
# or to get all interviews in status completed
suso_getAllInterviewQuestionnaire(
  workspace = "myworkspace",
  questID = "dee7705f-d611-4b12-9b97-2b8e5b80c4ea",
  version = 1
)
```

suso_getExportList	<i>Survey Solutions API call to list export processes</i>
--------------------	---

Description

Retrieves a list of export processes matching specified filters.

Usage

```
suso_getExportList(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  token = NULL,
  workspace = suso_get_api_key("workspace"),
  exportType = NULL,
  interviewStatus = NULL,
  questID = NULL,
  version = NULL,
```

```

    exportStatus = NULL,
    hasFile = NULL,
    limit = NULL,
    offset = NULL
  )

```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
workspace	server workspace, if nothing provided, defaults to primary
exportType	Format of export data: "Tabular", "STATA", "SPSS", "Binary", "DDI", "Parquet"
interviewStatus	Status of exported interviews: "All", "SupervisorAssigned", "InterviewerAssigned", "RejectedBySupervisor", "Completed", "ApprovedBySupervisor", "RejectedByHeadquarters", "ApprovedByHeadquarters"
questID	Questionnaire ID (GUID)
version	Questionnaire version
exportStatus	Status of export process: "Created", "Running", "Completed", "Fail", "Canceled"
hasFile	Logical, whether the export process has a file ready to download
limit	Maximum number of records to return
offset	Number of records to skip

Value

A data.table containing the list of export processes.

Examples

```

# List all export processes in workspace
exp_list <- susso_getExportList()

```

`suso_getExportProcess` *Survey Solutions API call to get detailed information about an export process*

Description

Retrieves status and details of a single export process by its ID.

Usage

```
suso_getExportProcess(  
  server = suso_get_api_key("susoServer"),  
  apiUser = suso_get_api_key("susoUser"),  
  apiPass = suso_get_api_key("susoPass"),  
  token = NULL,  
  workspace = suso_get_api_key("workspace"),  
  jobid = NULL  
)
```

Arguments

<code>server</code>	Survey Solutions server address
<code>apiUser</code>	Survey Solutions API user
<code>apiPass</code>	Survey Solutions API password
<code>token</code>	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
<code>workspace</code>	server workspace, if nothing provided, defaults to primary
<code>jobid</code>	Export process ID (integer)

Value

A data.table containing export process details.

Examples

```
# Get status of export process by jobid  
proc_info <- suso_getExportProcess(jobid = 123)
```

suso_getINT

*Survey Solutions API call for list of interviewers***Description**

Get list of all interviewers by supervisor id, or a list of all interviewers in the workspace.

Usage

```
suso_getINT(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  workspace = suso_get_api_key("workspace"),
  token = NULL,
  sv_id = NULL,
  bigTeam = FALSE
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	If workspace name is provide requests are made regarding this specific workspace, if no workspace is provided defaults to primary workspace.
token	If Survey Solutions server token is provided <i>usr</i> and <i>pass</i> will be ignored
sv_id	supervisor id, if NULL all interviewers in the workspace are returned, and a column with sv_id is added
bigTeam	(only if sv_id=NULL) if TRUE, requests will be performed sequential, and individual teams can be bigger than 100 if FALSE requests will be performed in parallel, but do not allow for teams larger 100 interviewers.

Value

An object of class UserClass (inheriting from data.table) containing the interviewers.

Examples

```
# Get all members for a single supervisor
suso_getINT(
  sv_id = "xxxx-xxxx-xxxx-xxx"
)
```

```
# Get all members in the workspace
suso_getINT()
```

suso_getINT_info	Survey Solutions API call for info on interviewers
------------------	--

Description

Get details of a single interviewer or retrieve the action log

Usage

```
suso_getINT_info(
    server = suso_get_api_key("susoServer"),
    apiUser = suso_get_api_key("susoUser"),
    apiPass = suso_get_api_key("susoPass"),
    int_id = NULL,
    workspace = suso_get_api_key("workspace"),
    token = NULL,
    log = FALSE,
    startDate = NULL,
    endDate = NULL
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
int_id	interviewer id
workspace	If workspace name is provide requests are made regarding this specific workspace, if no workspace is provided defaults to primary workspace.
token	If Survey Solutions server token is provided <i>usr</i> and <i>pass</i> will be ignored
log	If TRUE, then the audit log is returned
startDate	Start data for the period of interest, date must be provided as character in the format: YYYY-MM-DD, and will start at 00:00.00 server time of the specified day
endDate	End date for the period of interest, date must be provided ascharacter in the format: YYYY-MM-DD, and will end at 23:59:59 server time of the specified day. If not end date is provided, the current date will be used

Value

A data.table containing interviewer details or audit logs.

Examples

```
suso_getINT_info(  
  workspace = "myworkspace",  
  int_id = "xxxx-xxxx-xxxx-xxx"  
)
```

suso_getInterviewPDF	<i>Download interview PDF transcript</i>
----------------------	--

Description

Downloads the PDF transcript of a completed interview.

Usage

```
suso_getInterviewPDF(  
  server = suso_get_api_key("susoServer"),  
  apiUser = suso_get_api_key("susoUser"),  
  apiPass = suso_get_api_key("susoPass"),  
  workspace = suso_get_api_key("workspace"),  
  token = NULL,  
  intID = "",  
  path = NULL  
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace name
token	API token
intID	InterviewId (GUID)
path	Destination file path for saving the PDF. If NULL, saves to tempdir.

Value

The file path where the PDF was saved.

Examples

```
pdf_path <- suso_getInterviewPDF(intID = "11111111-2222-3333-4444-555555555555")
```

suso_getQuestDetails *Survey Solutions API call for questionnaire*

Description

suso_getQuestDetails implements all Questionnaire related API commands. It allows for different operation types, see details below for further clarification.

Usage

```
suso_getQuestDetails(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  workspace = suso_get_api_key("workspace"),
  token = NULL,
  questID = NULL,
  version = NULL,
  operation.type = c("list", "statuses", "structure", "interviews"),
  include_raw = FALSE,
  AssId = NULL,
  InterviewKey = NULL,
  errorsCount = NULL,
  errosCountFilter = c("lower", "higher", "equal"),
  interviewMode = c("CAPI", "CAWI"),
  notAnsweredCount = NULL,
  notAnsweredCountFilter = c("lower", "higher", "equal"),
  QuestionnaireVariable = NULL,
  ResponsibleName = NULL,
  responsibleRole = c("INTERVIEWER", "SUPERVISOR"),
  workStatus = c("All", "SupervisorAssigned", "InterviewerAssigned",
    "RejectedBySupervisor", "Completed", "ApprovedBySupervisor",
    "RejectedByHeadquarters", "ApprovedByHeadquarters"),
  supervisorName = NULL
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user

apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>usr</i> and <i>pass</i> will be ignored
questID	<i>QuestionnaireId</i> for which details should be exported
version	questionnaire version
operation.type	if <i>list</i> is specified a list of all questionnaires on the server. If <i>statuses</i> a vector of all questionnaire statuses. If <i>structure</i> is specified, it returns a list containing all questions, rosters etc. of the specific questionnaire, as well as all validations. If <i>interviews</i> is specified, all interviews for a specific questionnaire. See details below.
include_raw	logical; if TRUE, retains the raw JSON list column (. . JSON) in the question table when operation.type = "structure". Default is FALSE for clean, flat data tables.
AssId	Assignment ID (only required if operations.type is <i>interviews</i>)
InterviewKey	Interview key (only required if operations.type is <i>interviews</i>)
errorsCount	desired number of errors (only required if operations.type is <i>interviews</i>)
errosCountFilter	relational type for error counts, either smaller equal, equal or greater equal than the number specified in errorsCount, if not supplied defaults to lower equal than <i>interviews</i>) (only required if operations.type is <i>interviews</i>)
interviewMode	Interview mode (CAWI or CAPI) (only required if operations.type is <i>interviews</i>)
notAnsweredCount	number of unanswered questions (only required if operations.type is <i>interviews</i>)
notAnsweredCountFilter	relational type for unanswered question counts, either smaller equal, equal or greater equal than the number specified in notAnsweredCount, if not supplied defaults to lower equal than <i>interviews</i>) (only required if operations.type is <i>interviews</i>)
QuestionnaireVariable	the variable for the questionnaire (only required if operations.type is <i>interviews</i>)
ResponsibleName	Name of the person responsible (only required if operations.type is <i>interviews</i>)
responsibleRole	Role of the person responsible (only required if operations.type is <i>interviews</i>)
workStatus	of the interview (only required if operations.type is <i>interviews</i>)
supervisorName	Name of the supervisor of the responsible user (only required if operations.type is <i>interviews</i>)

Details

If *list* is selected, then list of questionnaires is returned.

If *statuses* is selected, a list of all available questionnaire statuses is returned (deprecated).

In case *structure* is chosen the return value is a list with four data.table elements:

- List element *q* contains all questions, rosters etc. with full metadata.

- List element *val* contains all validations.
- List element *v* is an alias for *val* for backward compatibility.
- List element *answers* contains categorical answer options, codes, and linked questions.

In this way it is straightforward to use the return value for questionnaire manuals and the likes.

In case interviews is selected, a list of all interviews for the specific questionnaire is returned.

Value

Depending on operation.type:

list A data.table listing all questionnaires on the server.

statuses A character vector of questionnaire statuses.

structure A list with four data.tables: q (questions/rosters metadata), val (validations), v (alias for val), and answers (categorical answer options and codes).

interviews A data.table listing interviews for the specified questionnaire.

Examples

```
# List all questionnaires on the server
q_list <- suso_getQuestDetails(operation.type = "list")

# Get questionnaire structure (questions, validations, answers)
q_struct <- suso_getQuestDetails(
  questID = q_list$QuestionnaireId[1],
  version = q_list$Version[1],
  operation.type = "structure"
)
```

susos_getQuestionsQuestionnaire

Survey Solutions API call for questions and responses from single questionnaire

Description

Returns all questions for a single questionnaire (ONLY if they contain responses). If you require all questions from any questionnaire on the server, use `suso_getQuestDetails(..., operation.type = "structure")`.

Usage

```
suso_getQuestionsQuestionnaire(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  workspace = suso_get_api_key("workspace"),
  token = NULL,
  questID = NULL,
  version = NULL
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
questID	Questionnaire ID (GUID)
version	Questionnaire version (numeric)

Value

A data.table listing questions with responses for the questionnaire.

Examples

```
suso_getQuestionsQuestionnaire(
  questID = "11111111-2222-3333-4444-555555555555",
  version = 1
)
```

suso_getStatsQuestionnaires

Survey Solutions API call for questionnaires with statistics

Description

Retrieves the list of questionnaires that have statistics data on the server.

Usage

```
suso_getStatsQuestionnaires(  
  server = suso_get_api_key("susoServer"),  
  apiUser = suso_get_api_key("susoUser"),  
  apiPass = suso_get_api_key("susoPass"),  
  workspace = suso_get_api_key("workspace"),  
  token = NULL  
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored

Value

A data.table listing questionnaires with statistics.

Examples

```
suso_getStatsQuestionnaires()
```

suso_getSV	<i>Survey Solutions API call for list of supervisors</i>
------------	--

Description

Gets list of supervisors

Usage

```
suso_getSV(  
  server = suso_get_api_key("susoServer"),  
  apiUser = suso_get_api_key("susoUser"),  
  apiPass = suso_get_api_key("susoPass"),  
  workspace = suso_get_api_key("workspace"),  
  token = NULL  
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	If workspace name is provide requests are made regarding this specific workspace, if no workspace is provided defaults to primary workspace.
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored

Value

An object of class `UserClass` (inheriting from `data.table`) containing the list of supervisors.

Examples

```
# receive all supervisors in the workspace
suso_getSV()
```

suso_getSV_info	<i>Survey Solutions API call for supervisor info</i>
-----------------	--

Description

Gets detailed info about a single supervisor by supervisor ID.

Usage

```
suso_getSV_info(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  sv_id = NULL,
  workspace = suso_get_api_key("workspace"),
  token = NULL
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
sv_id	supervisor ID (GUID)
workspace	server workspace name
token	API token

Value

A data.table containing supervisor details.

Examples

```
suso_getSV_info(  
  workspace = "myworkspace",  
  sv_id = "xxxx-xxxx-xxxx-xxx"  
)
```

suso_getUSR	<i>Survey Solutions API call for info on any user</i>
-------------	---

Description

Get any user’s info, by either providing the user id, email, or the user name

Usage

```
suso_getUSR(  
  server = suso_get_api_key("susoServer"),  
  apiUser = suso_get_api_key("susoUser"),  
  apiPass = suso_get_api_key("susoPass"),  
  user_id = NULL,  
  user_name = NULL,  
  user_email = NULL,  
  workspace = suso_get_api_key("workspace"),  
  token = NULL  
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
user_id	user id
user_name	user name
user_email	user email
workspace	If workspace name is provide requests are made regarding this specific workspace, if no workspace is provided defaults to primary workspace.
token	If Survey Solutions server token is provided <i>usr</i> and <i>pass</i> will be ignored

Value

A data.table containing user details.

Examples

```
suso_getUSR(  
  workspace = "myworkspace",  
  user_id = "xxxx-xxxx-xxxx-xxx"  
)
```

suso_getWorkspace

Survey Solutions API call for workspace information

Description

suso_getWorkspace allows you to get the list of workspaces, information about individual workspace names as well as workspace statuses. Workspaces as well as Workspaces information can only be accessed, if credentials are eligible. For more details please read <https://docs.mysurvey.solutions/headquarters/accounts/workspaces/>

Usage

```
suso_getWorkspace(  
  server = suso_get_api_key("susoServer"),  
  apiUser = suso_get_api_key("susoUser"),  
  apiPass = suso_get_api_key("susoPass"),  
  token = NULL,  
  workspace = NULL,  
  status = FALSE  
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
workspace	If workspace name is provide requests are made regarding this specific workspace
status	if status is <i>TRUE</i> workspace must be not NULL and status information about a specific workspace is requested

Value

A data.table containing workspace information or status.

Examples

```
# This assumes, that suso_PwCheck(workspace = "myworkspace") was
# sucessful

# shows all workspaces in the system AND the user has access to
suso_getWorkspace(
    status = FALSE)

# shows details for specific workspace myworkspace
suso_getWorkspace(
    workspace = "myworkspace",
    status = TRUE)
```

suso_globalNotice	<i>Survey Solutions API call for Headquarters Global Notice settings</i>
-------------------	--

Description

Allows getting, setting, and removing the global notice displayed in the Survey Solutions Headquarters application.

Usage

```
suso_globalNotice(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  token = NULL,
  action = c("get", "set", "delete"),
  message = NULL
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
action	Action to perform: "get" to retrieve the notice, "set" to update the notice, or "delete" to remove the notice
message	Character string containing the notice message. Required when action = "set".

Value

When action = "get", returns a character string with the current notice (or NULL if no notice is set). When action = "set" or "delete", returns TRUE invisibly on success.

Examples

```
# Get current global notice
suso_globalNotice(action = "get")

# Set a new global notice
suso_globalNotice(action = "set",
                  message = "Maintenance tonight at 22:00 UTC.")

# Delete the global notice
suso_globalNotice(action = "delete")
```

suso_keys

Survey Solutions API credentials

Description

(this function has been inspired by the googleway package <https://github.com/SymbolixAU/googleway>, an excellent package to use google geo-spatial API) Retrieves the list of Survey Solutions credentials that have been set.

Usage

```
suso_keys()
```

Value

A list of class suso_api containing current Survey Solutions credentials.

Examples

```
suso_keys()
```

suso_mapassign	<i>Assigns/Unassign a map to a user</i>
----------------	---

Description

Allows the user to assign/unassign a map to an interviewer to be used in CAPI data collection.

Usage

```
suso_mapassign(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  workspace = suso_get_api_key("workspace"),
  token = NULL,
  fileName = NULL,
  userName = NULL,
  assignUser = TRUE
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
fileName	the name of the map file on the server
userName	the name of the interviewer to whom the map will be assigned to
assignUser	if TRUE user will be assigned to map, if FALSE user will be removed from map

Value

A data.table containing the map assignment details (file name, user name, shape type, and import timestamp).

Examples

```
suso_mapassign(workspace = "myworkspace",
  fileName = "Lat9264Lon625_ALL.tif",
  userName = "INT0004")
```

suso_mapinfo

*Receive maps currently uploaded to the server***Description**

Allows the user to retrieve filtered or unfiltered map data including map details, like size, assigned users etc..

Usage

```
suso_mapinfo(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  workspace = suso_get_api_key("workspace"),
  token = NULL,
  fileName = NULL,
  importDateUtc = NULL,
  size = NULL,
  users = NULL,
  sortby_filename = NULL,
  sortby_importeddateutc = NULL,
  sortby_size = NULL,
  take = NULL,
  skip = NULL
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
fileName	name of the map on the server
importDateUtc	Import date
size	Size of the map
users	Users to whom the maps are assigned
sortby_filename	sort maps by file name, either ASC for ascending or DESC for descending
sortby_importeddateutc	sort maps by import date in utc, either ASC for ascending or DESC for descending
sortby_size	sort by map size, either ASC for ascending or DESC for descending
take	take the specified integer number of maps
skip	skip the first integer number of maps

Details

Attention: this uses the GraphQL API, not the REST API.

Value

Returns a data.table, with all the maps and additional information. If multiple users are assigned to a map, the table is expanded, such that there is one user per map.

Examples

```
suso_mapinfo(workspace = "myworkspace")
```

suso_mapreport	<i>Create a Map Report</i>
----------------	----------------------------

Description

Allows the user to create a map report with pre-specified parameters.

Usage

```
suso_mapreport(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  workspace = suso_get_api_key("workspace"),
  token = NULL,
  questID = NULL,
  version = NULL,
  variable = NULL,
  zoom = 1,
  clientMapWidth = 0,
  west = -180,
  east = 180,
  north = 90,
  south = -90,
  assignmentId = NULL,
  clientKey = NULL,
  createdAt = NULL,
  errorsCount = NULL,
  identifyingData = NULL,
  interviewMode = NULL,
  notAnsweredCount = NULL
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
questID	Questionnaire ID
version	Questionnaire version
variable	Variable(s) of interest
zoom	Zoom of the map report
clientMapWidth	width of the client map
west	coordinates for bounding box
east	coordinates for bounding box
north	coordinates for bounding box
south	coordinates for bounding box
assignmentId	Assignment ID
clientKey	Interview key
createdDate	Creation data of the interview
errorsCount	number of errors
identifyingData	Pre-loaded identifying data
interviewMode	Interview mode (CAWI or CAPI)
notAnsweredCount	number of unanswered questions

Value

A list containing the map report result.

Examples

```
suso_mapreport(
  workspace = "myworkspace",
  questID = "11111111-2222-3333-4444-555555555555",
  version = 1,
  variable = "gps_loc"
)
```

suso_mapupload	<i>Upload map to server</i>
----------------	-----------------------------

Description

Allows the user to upload a zip file of background maps and boundary files

Usage

```
suso_mapupload(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  workspace = suso_get_api_key("workspace"),
  token = NULL,
  path_to_zip = NULL
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
path_to_zip	either path to zip file or directory of files which will automatically be zipped as required

Details

Resources for upload must meet the requirements specified under: <https://docs.mysurvey.solutions/headquarters/mapsmanage/map-formats/>, Attention: this uses the GraphQL API, not the REST API.

Value

Returns a data.table, with information about the uploaded maps

Examples

```
# either upload the content of a folder
suso_mapupload(workspace = "myworkspace",
  path_to_zip = "./mapfiles/")

# or a zip file with the files included
suso_mapupload(workspace = "myworkspace",
```

```
path_to_zip = "../mapfiles/maps.zip")
```

suso_patchApproveInterview

Approve interviews either as supervisor or as headquarter.

Description

Allows you to approve interviews in supervisor or headquarters role, unapprove from headquarters status, as well as to provide a comment (i.e. reason).

Usage

```
suso_patchApproveInterview(
    server = suso_get_api_key("susoServer"),
    apiUser = suso_get_api_key("susoUser"),
    apiPass = suso_get_api_key("susoPass"),
    workspace = suso_get_api_key("workspace"),
    token = NULL,
    intID = "",
    HQ = FALSE,
    hqunapprove = FALSE,
    comment = "Well done!"
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
intID	the <i>InterviewId</i> of the interview.
HQ	if FALSE, approve as supervisor, if TRUE approve as headquarters
hqunapprove	if TRUE, unapprove from headquarters status (hqunapprove)
comment	comment which should be sent with the questionnaire

Details

For details please see: <https://docs.mysurvey.solutions/headquarters/interviews/survey-workflow/>

Value

A data.table containing the status of the approval operation.

Examples

```
# approve the interview as supervisor
suso_patchApproveInterview(
  workspace = "myworkspace",
  intID = "dee7705f-d611-4b12-9b97-2b8e5b80c4ea"
)
# approve the interview as headquarters
suso_patchApproveInterview(
  workspace = "myworkspace",
  intID = "dee7705f-d611-4b12-9b97-2b8e5b80c4ea",
  HQ = TRUE
)
```

suso_patchRejectInterview

Reject interviews either as supervisor or as headquarter.

Description

Allows you to reject interviews in supervisor or headquarters role as well as to provide a comment (i.e. reason) for the rejection.

Usage

```
suso_patchRejectInterview(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  workspace = suso_get_api_key("workspace"),
  token = NULL,
  intID = "",
  new_uid = NULL,
  HQ = FALSE,
  comment = "Please check errors and re-submit!"
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user

apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
intID	the <i>InterviewId</i> of the interview.
new_uid	if provided the interview will be rejected to a different user than the current one.
HQ	if FALSE, reject as supervisor, if TRUE rejected as headquarters
comment	comment which should be sent with the questionnaire

Details

For details please see: <https://docs.mysurvey.solutions/headquarters/interviews/survey-workflow/>

Value

A data.table containing the status of the rejection operation.

Examples

```
# reject the interview as supervisor
suso_patchRejectInterview(
  workspace = "myworkspace",
  intID = "dee7705f-d611-4b12-9b97-2b8e5b80c4ea"
)
# reject the interview as headquarters
suso_patchRejectInterview(
  workspace = "myworkspace",
  intID = "dee7705f-d611-4b12-9b97-2b8e5b80c4ea",
  HQ = TRUE
)
# reject the interview and provide a comment, so the interviewer knows about the problem
suso_patchRejectInterview(
  workspace = "myworkspace",
  intID = "dee7705f-d611-4b12-9b97-2b8e5b80c4ea",
  comment = "Too many errors, please check and re-submit!"
)
# reject the interview to a different user
suso_patchRejectInterview(
  workspace = "myworkspace",
  intID = "dee7705f-d611-4b12-9b97-2b8e5b80c4ea",
  new_uid = "3b0c6e09-d606-4914-9e20-2abc048d5bea"
)
```

`suso_PwCheck`*Utility function to check if credentials are correct*

Description

This function returns a 200 status code if credentials are correct and a 400 code otherwise.

Usage

```
suso_PwCheck(  
  server = suso_get_api_key("susoServer"),  
  apiUser = suso_get_api_key("susoUser"),  
  apiPass = suso_get_api_key("susoPass"),  
  workspace = suso_get_api_key("workspace"),  
  token = NULL  
)
```

Arguments

<code>server</code>	Survey Solutions Server
<code>apiUser</code>	API user
<code>apiPass</code>	API password
<code>workspace</code>	server workspace Name, if nothing provided, defaults to primary
<code>token</code>	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored

Details

If the app runs interactively, status is printed to the console, if it runs in a shiny app, a status notification will be shown, if option ‘suso.useshiny’ is TRUE.

Value

200 code if correct, 400 if incorrect.

Examples

```
suso_PwCheck(  
  server = "https://demo.mysurvey.solutions",  
  apiUser = "api_user",  
  apiPass = "password123"  
)
```

suso_questCriticalityLevel

Survey Solutions API call for questionnaire criticality level setting

Description

Gets or sets the criticality level setting for a specific questionnaire.

Usage

```
suso_questCriticalityLevel(
    server = suso_get_api_key("susoServer"),
    apiUser = suso_get_api_key("susoUser"),
    apiPass = suso_get_api_key("susoPass"),
    workspace = suso_get_api_key("workspace"),
    token = NULL,
    questID = NULL,
    version = NULL,
    level = NULL
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
questID	Questionnaire ID (GUID)
version	Questionnaire version (numeric)
level	Criticality level to set: "Unknown", "Ignore", "Warn", or "Block". If NULL (default), gets the current criticality level setting.

Value

When `level = NULL`, returns the current criticality level setting. When `level` is provided, returns TRUE invisibly upon successful update.

Examples

```
# Get current criticality level
suso_questCriticalityLevel(
  questID = "11111111-2222-3333-4444-555555555555",
  version = 1
)
```

```
# Set criticality level to Warn
suso_questCriticalityLevel(
  questID = "11111111-2222-3333-4444-555555555555",
  version = 1,
  level = "Warn"
)
```

suso_questRecordAudio *Survey Solutions API call for questionnaire audio recording setting*

Description

Gets or sets the audio recording setting for a specific questionnaire.

Usage

```
suso_questRecordAudio(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  workspace = suso_get_api_key("workspace"),
  token = NULL,
  questID = NULL,
  version = NULL,
  enabled = NULL
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
workspace	server workspace, if nothing provided, defaults to primary
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
questID	Questionnaire ID (GUID)
version	Questionnaire version (numeric)
enabled	Logical. If provided (TRUE or FALSE), updates the audio recording setting. If NULL (default), retrieves the current audio recording setting.

Value

When enabled = NULL, returns a list with Enabled (logical) and AudioAuditScope (character vector). When enabled is provided, returns TRUE invisibly upon successful update.

Examples

```
# Get current audio recording setting
suso_questRecordAudio(
  questID = "11111111-2222-3333-4444-555555555555",
  version = 1
)
# Enable audio recording
suso_questRecordAudio(
  questID = "11111111-2222-3333-4444-555555555555",
  version = 1,
  enabled = TRUE
)
```

suso_set_assignments *Survey Solutions API call for assignment manipulation*

Description

suso_set_assignments allows to (re-)assign, change limits or audio recording settings, as well as archiving/unarchive, downsizing, changing status/target area, and closing assignments.

Usage

```
suso_set_assignments(
  server = suso_get_api_key("susoServer"),
  apiUser = suso_get_api_key("susoUser"),
  apiPass = suso_get_api_key("susoPass"),
  token = NULL,
  workspace = suso_get_api_key("workspace"),
  AssId = NULL,
  payload = NULL,
  operations.type = c("recordAudio", "archive", "assign", "changeQuantity", "close",
    "unarchive", "downsize", "changeStatus", "changeTargetArea")
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
token	If Survey Solutions server token is provided <i>apiUser</i> and <i>apiPass</i> will be ignored
workspace	If workspace name is provide requests are made regarding this specific workspace
AssId	the assignment id for which the change is required

payload requirements depend on the operations type. See details below.

operations.type specifies the desired operation, one of `recordAudio`, `archive`, `unarchive`, `assign`, `changeQuantity`, `close`, `downsize`, `changeStatus`, `changeTargetArea`.

Details

If `operations.type` is *recordAudio*, `TRUE/FALSE` is required as payload. If it is *archive*, *unarchive*, *close*, or *downsize*, no payload is required. If it is *assign* the payload must be the user ID (UUID) or username of the new responsible person (character, or `data.frame/list` with `Responsible`). If it is *changeQuantity* the payload must be the new integer number of assignments (-1 for unlimited). If it is *changeStatus* the payload must be the new status string (e.g. "Closed", "Deleted") or a named list with `Status` and optional `Comment`. If it is *changeTargetArea* the payload must be the new target area string.

Value

Returns an S3 object of `assignmentClass`

Examples

```
# (re-)assign existing assignment by UUID, username, or data.frame
asslist<-suso_set_assignments(
  workspace = "myworkspace",
  AssId = 10,
  payload = "43f3d2bd-7959-4706-97ae-2653b5685c9e",
  operations.type = "assign"
)

asslist<-suso_set_assignments(
  workspace = "myworkspace",
  AssId = 10,
  payload = data.frame(Responsible = "interviewer01"),
  operations.type = "assign"
)

# downsize assignment
asslist<-suso_set_assignments(
  workspace = "myworkspace",
  AssId = 10,
  operations.type = "downsize"
)
```

Description

Sets API credentials so it's available for all API calls. See details

Usage

```
suso_set_key(  
    suso_server = "",  
    suso_user = "",  
    suso_password = "",  
    workspace = NULL,  
    suso_token = ""  
)
```

Arguments

<code>suso_server</code>	Survey Solutions server address
<code>suso_user</code>	Survey Solutions API user
<code>suso_password</code>	Survey Solutions API password
<code>workspace</code>	server workspace Name, if nothing provided, defaults to primary
<code>suso_token</code>	If Survey Solutions server token is provided <i>suso_user</i> and <i>suso_password</i> will be ignored

Details

Use `suso_set_key` to make API keys available for all the `suso_` functions, so you don't need to specify the credentials parameter within those functions. The server address can be provided with or without `https:\` suffix, nevertheless if it is missing, then the suffix will be added. For testing purposes it also allows for `http` connections, however for publicly accessible servers we do not recommend unencrypted connections.

In case *suso_token* is provided, only token authentication will be attempted. For details on token authentication in Survey Solutions please see <https://docs.mysurvey.solutions/headquarters/accounts/token-based-authentication/>.

Value

Invisible NULL.

Examples

```
suso_set_key(  
    suso_server = "https://demo.mysurvey.solutions",  
    suso_user = "api_user",  
    suso_password = "password123",  
    workspace = "primary"  
)  
suso_clear_keys()
```

susosetmaxparreq	<i>Convenience Function to modify the maximum number of parallel requests option</i>
------------------	--

Description

This function allows to modify the maximum number of parallel requests 'susosetmaxparreq' in a single call. It also checks the maximum number of parallel connections, and if lower than the number of requests maxreq then also sets the 'susosetmaxparreq' option to maxreq.

Usage

```
susosetmaxparreq(maxreq = 100)
```

Arguments

maxreq	set the number of parallel requests for all functions which
--------	---

Value

Invisible NULL.

Examples

```
# Set maximum parallel requests to 50
susosetmaxparreq(50)
# Reset back to default
susosetmaxparreq(100)
```

susosetpwcheckmess	<i>Convenience functions to modify user notification text</i>
--------------------	---

Description

These functions allow the user to customize certain notifications and messages when used in shiny apps.

Usage

```
susosetpwcheckmess(messsucc = NULL, messfail = NULL)
```

```
susosetprogmess(mess = NULL)
```

Arguments

mess_succ	set the success message for suso_PwCheck when used in shiny app.
mess_fail	set the success message for suso_PwCheck when used in shiny app.
mess	set the message for the export progress bars when used in shiny app, used in suso_export and suso_export_paradata

Value

always invisible TRUE.

Examples

```
# Set custom notification messages for shiny app
suso_set_pwcheck_mess(mess_succ = "Connected successfully!", mess_fail = "Authentication failed!")
suso_set_prog_mess(mess = "Downloading data, please wait...")
```

suso_set_workspace	<i>Convenience function to switch workspace</i>
--------------------	---

Description

Sets the workspace only, but leaves all other credentials the same.

Usage

```
suso_set_workspace(workspace = NULL)
```

Arguments

workspace	server workspace Name (not the display name), if nothing provided, defaults to primary
-----------	--

Details

Use `suso_set_workspace` to make the desired workspace available for all the `suso_` functions, so you don't need to specify the workspace parameter within those functions. The function also checks if the workspace name is correct, and the user with the current credentials is authorized. If the workspace requires different credentials, then use [suso_set_key](#) again.

Value

invisibly TRUE if successful.

Examples

```
# switch to workspace "windows"
susos_set_workspace("windows")

# switch to primary (default) workspace
susos_set_workspace()
```

susos_updateWorkspace *Update workspace details*

Description

Updates the display name of a workspace. Accessible only to administrator.

Usage

```
susos_updateWorkspace(
  server = susos_get_api_key("susosServer"),
  apiUser = susos_get_api_key("susosUser"),
  apiPass = susos_get_api_key("susosPass"),
  token = NULL,
  workspace = NULL,
  displayName = NULL
)
```

Arguments

server	Survey Solutions server address
apiUser	Survey Solutions API user
apiPass	Survey Solutions API password
token	API token
workspace	Name of the workspace to update
displayName	New display name for the workspace

Value

A data.table indicating the update status.

Examples

```
susos_updateWorkspace(workspace = "my_ws", displayName = "Renamed Workspace")
```

UserClass

*User Classes & Methods***Description**

Users class extends the data.table class to include relevant methods

Usage

```
UserClass(x, ...)
```

Arguments

x	list returned by api call
...	additional attributes to be added to the assignmentClass

Value

An object of class UserClass (inheriting from data.table)

Examples

```
usr_list <- list(Users = data.frame(UserName = "interviewer1", Role = "Interviewer"))
uc <- UserClass(usr_list)
is.UserClass(uc)
```

Index

- * **suso-customization-options**
 - suso_set_pwcheck_mess, 74
- assignmentClass, 4
- boxplot_summary, 4
- calendarevent, 5
- exportClass, 7
- getinfo, 8
- is.assignmentClass, 9
- is.exportClass, 10
- is.UserClass, 10
- OlsonNames, 6
- plan, 14
- req_perform_parallel, 22
- sf, 31
- summaryTable, 11
- summaryTable.UserClass, 13
- suso.options, 14
- suso_archUSR, 15
- suso_assignInterview, 16
- suso_assignWorkspace, 17
- suso_cancelExport, 19
- suso_clear_keys, 20
- suso_commentInterview, 21
- suso_createASS, 22
- suso_createCALEV (calendarevent), 5
- suso_createUSER, 23
- suso_createWorkspace, 24
- suso_delCALEV (calendarevent), 5
- suso_deleteInterview, 26
- suso_deletemap, 27
- suso_deleteWorkspace, 28
- suso_enableWorkspace, 28
- suso_export, 15, 29, 75
- suso_export_paradata, 32, 75
- suso_get_api_key, 35
- suso_get_assignments, 36
- suso_get_default_key, 38
- suso_get_stats, 39
- suso_get_stats_interview, 40
- suso_getAllAnswerInterview, 41
- suso_getAllHistoryInterview, 42
- suso_getAllInterviewQuestionnaire, 43
- suso_getExportList, 44
- suso_getExportProcess, 46
- suso_getINT, 47
- suso_getINT_info, 48
- suso_getInterviewPDF, 49
- suso_getQuestDetails, 44, 50
- suso_getQuestionsQuestionnaire, 52
- suso_getStatsQuestionnaires, 53
- suso_getSV, 54
- suso_getSV_info, 55
- suso_getUSR, 56
- suso_getWorkspace, 57
- suso_globalNotice, 58
- suso_keys, 59
- suso_mapassign, 60
- suso_mapinfo, 61
- suso_mapreport, 62
- suso_mapupload, 64
- suso_patchApproveInterview, 65
- suso_patchRejectInterview, 66
- suso_PwCheck, 14, 68, 75
- suso_questCriticalityLevel, 69
- suso_questRecordAudio, 70
- suso_set_assignments, 71
- suso_set_key, 72, 75
- suso_set_maxpar_req, 74
- suso_set_prog_mess
 - (suso_set_pwcheck_mess), 74

suso_set_pwcheck_mess, [74](#)
suso_set_workspace, [75](#)
suso_updateCALEV (calendarevent), [5](#)
suso_updateWorkspace, [76](#)

UserClass, [77](#)

withProgress, [15](#)