Package 'carboseqsiwaa'

October 24, 2024

Title Commands Siwaa tool 'CarboSeqSimulator'

Version 0.0.0.9000

Description Provides methods to command Siwaa tool 'CarboSeqSimulator'.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Imports httr,
jsonlite,
rjson

Contents

allDataReady *All Datasets of a collection are ready*

Description

All Datasets of a collection are ready

Usage

```
allDataReady(dataset_collection_info)
```

Arguments

dataset_collection_info	Dataset collection information.

Value

Bolean All contained datasets ready

checkSiwaaCom *Check communication with Siwaa*

Description

Check communication with Siwaa

Usage

```
checkSiwaaCom(server, api_key)
```

Arguments

server	Siwaa server url
api_key	Available Siwaa API KEY

Value

check_ok Boolean

report Details in case of check_ok valuing FALSE

`clearHistory` *Clear the history containing the dataset*

Description

Clearing action is purge if 'purge' is TRUE, and else delete

Usage

```
clearHistory(env = env, data_info, purge = TRUE)
```

Arguments

`env`	Siwaa environment
`data_info`	Data information (contains history id).
`purge`	Boolean to choose between purge and delete

Value

history_id History Id

history_name History name

deleted Boolean

purged Boolean

message

`ControlCommunicationWithSiwaa`
Control Communication with Siwaa by API

Description

Control Communication with Siwaa by API

Usage

```
ControlCommunicationWithSiwaa(env)
```

Arguments

`env`	Siwaa environment

Value

check_ok Boolean

report Details in case of check_ok valuing FALSE

createDatasetCollection *Create a dataset collection, into a history, from a dataset list*

Description

Create a dataset collection, into a history, from a dataset list

Usage

```
createDatasetCollection(
  history_id,
  data_info_list,
  dataset_collection_name,
  server,
  api_key
)
```

Arguments

history_id	History id
data_info_list	dataset information list
dataset_collection_name	dataset collection name
server	Siwaa server url
api_key	Available Siwaa API KEY

Value

dataset_collection_info created dataset collection information

createHistory *Create a history*

Description

Create a history

Usage

```
createHistory(history_name, server, api_key)
```

Arguments

history_name	History name
server	Siwaa server url
api_key	Available Siwaa API KEY

Value

history_id Id of created history

---

createSiwaaEnv *Create siwaaenv*

Description

Create and initialize a Siwaa environment

Usage

```
createSiwaaEnv(
  SERVER = "https://siwaa.toulouse.inrae.fr",
  CARBOSEQ_TOOL_ID =
   "toolshed-siwaa.toulouse.inrae.fr/repos/patrick_chabrier/carboseq_s/CarboSeqSimulator/1.0.8",
  CARBOSEQ_WORKFLOW_ID = "32c9385265d8ab1a",
  API_KEY = Sys.getenv("CARBOSEQ_SIWAA_TOKEN"),
  INPUT_DIR = "../INPUTS/40Sites/csvdir",
  OUTPUT_DIR = "../OUTPUTS",
  HISTORY_ID = "4f93aea6ec2af23e",
  HISTORY_NAME = "CarboSeq [CarboSeqSimulator.R]",
  DELAY = 5,
  DURATION_MAX = 600,
  INVOCATION_DELAY = 10,
  INVOCATION_DURATION_MAX = 600,
  VERBOSE = FALSE
)
```

Arguments

SERVER	Siwaa server url (mandatory)
CARBOSEQ_TOOL_ID	'CarboSeqSimulator' tool id (default value for parameter of some methods)
CARBOSEQ_WORKFLOW_ID	'CarboSeqRun' workflow id (default value for parameter of some methods)
API_KEY	Available Siwaa API KEY (mandatory)
INPUT_DIR	Input folder containing the .csv input files (will be zipped) (default value for parameter of some methods)

OUTPUT_DIR	Folder destination of downloaded result file (default value for parameter of some methods)
HISTORY_ID	An existing history id (default value for parameter of some methods)
HISTORY_NAME	History name for a created history (default value for parameter of some methods)
DELAY	Delay (dataset survey). Unit : second. Default parameter value for 'waitSimsResults' method.
DURATION_MAX	Duration max (dataset survey). Unit : second. Default parameter value for 'waitSimsResults' method.
INVOCATION_DELAY	Delay (invocation survey). Unit : second. Default parameter value for some methods surveying workflow invocation state evolution ('runWorkflow', 'waitWorkflowOutputDataIdent').
INVOCATION_DURATION_MAX	Duration max (invocation survey). Unit : second. Default parameter value for some methods surveying workflow invocation state evolution ('waitWorkflowOutputDataIdent').
VERBOSE	Boolean to choose 'verbose' or 'silent' mode (default value for parameter of some methods)

Details

siwaaenv contains some mandatory constants (SERVER, API_KEY) that must be verified and modified if necessary.

siwaaenv contains some constants (OUTPUT_DIR, CARBOSEQ_TOOL_ID, CARBOSEQ_WORKFLOW_ID, DURATION_MAX...) used as default values for some methods parameters : the user can choose to update them, or to give his own values by methods parameters. Example : setSiwaaEnv(..., DURATION_MAX=60) vs waitSimsResults(..., duration_max=60)

The method to update siwaaenv is setSiwaaEnv.

siwaaenv will be required by many methods as 'env' parameter.

Value

Siwaa environment

`dateTime4History` *Return a string with date and time*

Description

Return a string with date and time

Usage

```
dateTime4History()
```

Value

a string

`deleteHistory` *Purge a history if 'purge' is TRUE, and else delete it*

Description

Purge a history if 'purge' is TRUE, and else delete it

Usage

```
deleteHistory(history_id, purge = TRUE, server, api_key)
```

Arguments

`history_id`	History id
`purge`	Boolean to choose between purge and delete
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY

Value

history_info History information

`downloadDatasetCollection`
Download as file a dataset collection

Description

Return downloaded file path.

Usage

```
downloadDatasetCollection(dataset_collection_info, output_dir, server, api_key)
```

Arguments

`dataset_collection_info`	Dataset collection information
`output_dir`	Path of folder destination of downloaded file
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY

Value

Downloaded file path

downloadInMemory *Download in memory a dataset*

Description

The dataset must be ready to be able to be downloaded.

Usage

```
downloadInMemory(data_info, server, api_key)
```

Arguments

data_info	Data information
server	Siwaa server url
api_key	Available Siwaa API KEY

Value

downloaded data value

getApiKeyValue *Return env API_KEY value*

Description

Return env API_KEY value

Usage

```
getApiKeyValue(env)
```

Arguments

env	Siwaa environment

Value

env API_KEY value

`getDataInfo` *Get data information*

Description

Get information of a dataset

Usage

```
getDataInfo(data_id, server, api_key)
```

Arguments

`data_id`	Data id
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY

Details

The data information (data_info) contains many fields among which :

- its state ('state') allowing to follow process progression
- its own id ('id'), history id ('history_id'), job id ('creating_job').
- url ('url') to get again data information (refresh).
- url ('download_url') to download data.
- some data file information ('name', 'file_ext', 'file_size', 'misc_info'...) that will be used when naming the downloaded file.

Value

data_info Data information.

`getDatasetCollectionIdList`
List of ids of Datasets of a collection

Description

List of ids of Datasets of a collection

Usage

```
getDatasetCollectionIdList(dataset_collection_info)
```

Arguments

`dataset_collection_info`
: Dataset collection information.

Value

Id list of collection datasets

`getDatasetCollectionInfo`
Get dataset collection information

Description

Get information of a dataset collection of a history

Usage

```
getDatasetCollectionInfo(dataset_collection_id, history_id, server, api_key)
```

Arguments

`dataset_collection_id`
: Dataset collection id

`history_id`
: History id

`server`
: Siwaa server url

`api_key`
: Available Siwaa API KEY

Details

The dataset collection information (dataset_collection_info) contains many fields among which :

- its own id ('id'), history id ('history_id'), its own name ('name')
- its contained datasets ('elements') and their number ('element_count')
- url ('url') to get again dataset collection information (refresh).

Value

dataset_collection_info Dataset collection information.

`getDatasetCollectionJobIdList`
List of job ids of Datasets of a collection

Description

List of job ids of Datasets of a collection

Usage

```
getDatasetCollectionJobIdList(dataset_collection_info, server, api_key)
```

Arguments

`dataset_collection_info`	Dataset collection information.
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY

Value

Job Id list of collection datasets

`getDatasetCollectionStateList`
List of states of Datasets of a collection

Description

List of states of Datasets of a collection

Usage

```
getDatasetCollectionStateList(dataset_collection_info)
```

Arguments

`dataset_collection_info`	Dataset collection information.

Value

State list of collection datasets

`getInvocationInfo` *Get invocation information*

Description

Get information of a workflow invocation

Usage

```
getInvocationInfo(invocation_id, server, api_key)
```

Arguments

`invocation_id`	Invocation id
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY

Details

The workflow invocation information represents the scheduling of workflow. It may be sparse at first (missing inputs and invocation steps) and will become more populated as the workflow is actually scheduled.

The workflow invocation information (invocation_info) contains many fields among which :

- its state ('state') allowing to follow invocation progression.
- its own id ('id'), history id ('history_id'), workflow id ('workflow_id').
- its steps ('steps') with for each step : 'id', 'job_id', 'state'...
- inputs ('inputs', 'input_step_parameters'), outputs ('outputs')...

Value

invocation_info Invocation information.

`getInvocationOutputDataId`
Get the output data Id from invocation information

Description

The output data can be identified only if the workflow invocation is in 'scheduled' terminal state.

Usage

```
getInvocationOutputDataId(invocation_info)
```

Arguments

invocation_info	Workflow invocation information

Details

Return the output data id (in case of 'scheduled' state), that will be useful to survey/follow progression and then download the result file (when ready). In any case of state other than 'scheduled', the output data id is invalid (value "").

Important : The label value ('main_output') of the output data into 'CarboSeqRun' workflow (cf 'outputs') will be used here to identify it.

Value

output_data_id Id of output data Value "" if unidentified output data

---

getInvocationReport *Get invocation report*

Description

Get report for a workflow invocation

Usage

```
getInvocationReport(invocation_id, server, api_key)
```

Arguments

invocation_id	Invocation id
server	Siwaa server url
api_key	Available Siwaa API KEY

Details

TODO : not filtered for the moment (only some fields could be kept)

Value

invocation_report Invocation report.

`getInvocationStepInfo` *Get an invocation step information*

Description

Get information (details) of a particular workflow invocation step

Usage

```
getInvocationStepInfo(invocation_id, step_id, server, api_key)
```

Arguments

`invocation_id`	Invocation id
`step_id`	Invocation step id
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY

Details

TODO : not filtered for the moment (only some fields could be kept)

Value

invocation_step_info Invocation step information.

`getInvocationStepJobsSummaryInfo`
Get invocation step jobs summary information

Description

Get a detailed summary information of an invocation, listing all jobs with their job IDs and current states.

Usage

```
getInvocationStepJobsSummaryInfo(invocation_id, server, api_key)
```

Arguments

`invocation_id`	Invocation id
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY

Value

invocation_step_jobs_summary_info Invocation step jobs summary information.

`getInvocationSummaryInfo` *Get invocation summary information*

Description

Get a summary information of an invocation, stating the number of jobs which succeed, which are paused and which have errored.

Usage

```
getInvocationSummaryInfo(invocation_id, server, api_key)
```

Arguments

`invocation_id`	Invocation id
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY

Value

invocation_summary_info Invocation summary information.

`getJobInfo` *Get job information*

Description

TODO : not filtered for the moment (only some fields could be kept)

Usage

```
getJobInfo(job_id, server, api_key)
```

Arguments

`job_id`	Job id
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY

Value

job_info Job information.

`getJobMetrics` *Get job metrics*

Description

TODO : not filtered for the moment (only some fields could be kept)

Usage

```
getJobMetrics(job_id, server, api_key)
```

Arguments

`job_id`	Job id
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY

Value

job_metrics Job metrics.

`getJobReportList` *Get list of job reports*

Description

Get list of job reports

Usage

```
getJobReportList(env, job_id_list)
```

Arguments

`env`	Siwaa environment
`job_id_list`	List of job ids

Value

job_report_list List of job reports (information and metrics)

`getServerValue` *Return env SERVER value*

Description

Return env SERVER value

Usage

```
getServerValue(env)
```

Arguments

`env`	Siwaa environment

Value

env SERVER value

`getSiwaaEnv` *Return value of name from env*

Description

Return value of name from env

Usage

```
getSiwaaEnv(env, name)
```

Arguments

`env`	Siwaa environment
`name`	name into env

Value

env name value

`getVerboseValue` *Return env VERBOSE value*

Description

Return env VERBOSE value

Usage

```
getVerboseValue(env)
```

Arguments

`env`	Siwaa environment

Value

env VERBOSE value

`idsInfoDatasetCollection`
Ids information about a dataset collection content

Description

Ids information about a dataset collection content

Usage

```
idsInfoDatasetCollection(env, dataset_collection_info)
```

Arguments

`env`	Siwaa environment
`dataset_collection_info`	Dataset collection information

Value

id list and job id list of collection datasets

`invokeCarboseqworkflow` *Invoke 'CarboSeqRun' workflow*

Description

Invoke 'CarboSeqRun' workflow into a history with an input data.

Usage

```
invokeCarboseqworkflow(
  input_data_info,
  chunck_size,
  workflow_id,
  server,
  api_key
)
```

Arguments

`input_data_info`	Input data information. input_data_info contains input data id and history id.
`chunck_size`	Input parameter (Integer given as str). Value should be adapted to input_data_info size.
`workflow_id`	'CarboSeqRun' workflow id
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY

Details

Return the workflow invocation id, that will be useful to survey/follow progression of the workflow invocation (step before being able to identify the output data id, then survey/follow progression and finally download the result file (when ready)).

Important : specific of 'CarboSeqRun' workflow. Code written for the case of a workflow with 'same' main inputs than it.

2 inputs have to be prepared : 'main_input' dataset (zip) and 'chunck_size' parameter (integer)

Note : Code allowing to identify inputs indexes : url = paste(SERVER, "/api/workflows/", WORKFLOW_ID, sep="") headers=c("x-api-key"=API_KEY) r <- httr::GET(url=url, httr::add_headers(.headers=headers)) wf = httr::content(r) #print(wf) #print(wf$'inputs') # => indexes main_input:'0' chunck_size:'1'

Value

invocation_id Id of workflow invocation

`isDataInATerminalState` *Define if a dataset is or not in a terminal state*

Description

Data state : a data is in some non-terminal states (along process) then in a terminal state (at the end) :

- terminal states : see TERMINAL_STATES list
- non-terminal states are : 'new', 'upload', 'queued', 'running', 'paused', 'setting_metadata'

Usage

```
isDataInATerminalState(data_info)
```

Arguments

`data_info`	Data information.

Value

Boolean

`isDataReady` *Define if a dataset is ready or not*

Description

Data ready criteria : 'state':'ok'

Usage

```
isDataReady(data_info)
```

Arguments

`data_info`	Data information.

Value

Boolean

`isInvocationInATerminalState` *Define if a workflow invocation is or not in a terminal state*

Description

Invocation state : a workflow invocation is in some non-terminal states then in some terminal states :

- Invocation terminal states : see INVOCATION_TERMINAL_STATES list
- Invocation non-terminal states are : 'cancelling', 'new', 'ready'

Usage

```
isInvocationInATerminalState(invocation_info)
```

Arguments

`invocation_info`
: Invocation information.

Value

Boolean

`isInvocationScheduled` *Define if a workflow invocation is scheduled or not*

Description

Invocation scheduled criteria : 'state':'scheduled'

Usage

```
isInvocationScheduled(invocation_info)
```

Arguments

`invocation_info`
: Invocation information.

Value

Boolean

`jobId` *Returns job id of a dataset of a history*

Description

Returns job id of a dataset of a history

Usage

```
jobId(data_info)
```

Arguments

`data_info`	Data information (containing job id ('creating_job'))

Value

Job id

`launchWorkflow` *Launch simulations by workflow*

Description

Launch 'CarboSeqRun' workflow ('workflow_id') invocation into a history with an input data and return immediatly. Does not wait for Workflow invocation to be in a terminal state.

Usage

```
launchWorkflow(env, input_data_info, chunck_size, workflow_id, v)
```

Arguments

`env`	Siwaa environment
`input_data_info`	Input data information.
`chunck_size`	Input parameter (Integer given as str). Value should be adapted to input_data_info size.
`workflow_id`	'CarboSeqRun' workflow id
`v`	Boolean to choose 'verbose' or 'silent' mode

Details

To then survey workflow invocation state evolution (wait for a terminal state) : 'waitWorkflowOutputDataIdent' method can be used.

Value

invocation_id Id of workflow invocation

message

`makeDatasetCollection` *Make a dataset collection*

Description

Make a dataset collection

Usage

```
makeDatasetCollection(env, data_info_list, dataset_collection_name)
```

Arguments

`env`	Siwaa environment
`data_info_list`	List of dataset information
`dataset_collection_name`	Name of dataset collection to be created

Value

Dataset collection information

`memResults` *Download in memory a dataset*

Description

Before downloading, verify that the dataset is ready. Return downloaded data value ('cr_ok' TRUE case) or "" if unsuccess ('cr_ok' FALSE case)

Usage

```
memResults(env, data_info)
```

Arguments

`env`	Siwaa environment
`data_info`	Data information

Value

cr_ok Boolean

value Downloaded data value or ""

data_state data state

message

`refreshDatasetCollectionInfo` *Refresh dataset collection information*

Description

Refresh dataset collection information

Usage

```
refreshDatasetCollectionInfo(dataset_collection_info, server, api_key)
```

Arguments

`dataset_collection_info`	Dataset collection information.
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY

Value

dataset_collection_info Dataset collection information (refreshed).

`runAllSims` *Run all simulations*

Description

Launch 'CarboSeqSimulator' tool ('tool_id') execution into a history for each input data.

Usage

```
runAllSims(env, input_data_info_list, tool_id)
```

Arguments

`env`	Siwaa environment
`input_data_info_list`	Input data information list.
`tool_id`	'CarboSeqSimulator' tool id (determining tool version)

Value

output_data_info_list Output data information list, each output_data_info containing :

- output_data_info$history_id
- output_data_info$creating_job allowing to know tool_id, input_data_id

---

`runCarboseqsimulator` *Run 'CarboSeqSimulator' tool*

Description

Run 'CarboSeqSimulator' tool into a history with an input data.

Usage

```
runCarboseqsimulator(input_data_info, tool_id, server, api_key)
```

Arguments

`input_data_info`	Input data information. input_data_info contains input data id and history id.
`tool_id`	'CarboSeqSimulator' tool id
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY

Details

Return the output data id, that will be useful to survey/follow progression and to download the result file (when ready).

Important : specific of 'CarboSeqSimulator' tool. Code written for the case of a tool with ONLY ONE input and ONLY ONE output.

Value

output_data_id Id of output dataset

runSims — *Run simulations by tool*

Description

Launch 'CarboSeqSimulator' tool ('tool_id') execution into a history with an input data.

Usage

```
runSims(env, input_data_info, tool_id, v)
```

Arguments

env	Siwaa environment
input_data_info	Input data information.
tool_id	'CarboSeqSimulator' tool id (determining tool version)
v	Boolean to choose 'verbose' or 'silent' mode

Value

output_data_info Output data information, containing :

- output_data_info$history_id
- output_data_info$creating_job allowing to know tool_id, input_data_id

message

runWorkflow — *Run simulations by workflow*

Description

Launch 'CarboSeqRun' workflow ('workflow_id') execution (invocation) into a history with an input data. Then get and return the output data information, after having waited (infinite loop) for invocation to be scheduled (in order to be able to access to invocation outputs information).

Usage

```
runWorkflow(env, input_data_info, chunck_size, workflow_id, delay, v)
```

Arguments

env	Siwaa environment
input_data_info	Input data information. input_data_info contains input data id and history id.
chunck_size	Input parameter (Integer given as str). Value should be adapted to input_data_info size.
workflow_id	'CarboSeqRun' workflow id
delay	Delay (invocation survey). Unit : second.
v	Boolean to choose 'verbose' or 'silent' mode

Details

The infinite loop wait for workflow invocation to reach a terminal state. It refresh invocation information, with a 'delay' between 2 requests. If the terminal state is 'scheduled', then it is possible to access to invocation outputs information, output_data_identified is TRUE. If it is a terminal state other than 'scheduled', then it is not possible to access to invocation outputs information, output_data_identified is FALSE.

It is possible to set the survey step between 2 requests ('delay').

Value

output_data_info Output data information, containing output_data_info$history_id, output_data_info$creating_job...

output_data_identified Boolean to know if output_data_info valid

terminal_state Boolean,

invocation_scheduled Boolean,

invocation_info Workflow invocation information

message

---

saveResults *Download as file a dataset*

Description

The dataset must be ready to be able to be downloaded. Return downloaded file path or "" if unsuccess (data not ready).

Usage

```
saveResults(env, data_info, output_dir)
```

Arguments

env	Siwaa environment
data_info	Data information
output_dir	Path of folder destination of downloaded file

Value

Downloaded file path (valuing "" if download failed)

message

---

saveResultsAllTogether

Download as one .zip file a dataset collection

Description

The .zip file contains the .RData result files of ready output datas : while calling the method. while calling the method.

Usage

```
saveResultsAllTogether(env, dataset_collection_info, output_dir)
```

Arguments

env	Siwaa environment
dataset_collection_info	Dataset collection information
output_dir	Path of folder destination of downloaded file

Value

Downloaded file path

setSiwaaEnv *Update Siwaa environment with values of given parameters*

Description

Update Siwaa environment with values of given parameters

Usage

```
setSiwaaEnv(
  env,
  SERVER,
  CARBOSEQ_TOOL_ID,
  CARBOSEQ_WORKFLOW_ID,
  API_KEY,
  INPUT_DIR,
  OUTPUT_DIR,
  HISTORY_ID,
  HISTORY_NAME,
  DELAY,
  DURATION_MAX,
  INVOCATION_DELAY,
  INVOCATION_DURATION_MAX,
  VERBOSE
)
```

Arguments

env	Siwaa environment
SERVER	Siwaa server url
CARBOSEQ_TOOL_ID	'CarboSeqSimulator' tool id
CARBOSEQ_WORKFLOW_ID	'CarboSeqRun' workflow id
API_KEY	Available Siwaa API KEY
INPUT_DIR	Input folder containing the .csv input files
OUTPUT_DIR	Folder destination of downloaded result file
HISTORY_ID	An existing history id
HISTORY_NAME	History name for a created history
DELAY	Delay (dataset survey). Unit : second.
DURATION_MAX	Duration max (dataset survey). Unit : second.
INVOCATION_DELAY	Delay (invocation survey). Unit : second.
INVOCATION_DURATION_MAX	Duration max (invocation survey). Unit : second.
VERBOSE	Boolean to choose 'verbose' or 'silent' mode.

Value

Siwaa environment

`simpleBatchRunSim` *Launch a plan of simulation on a cluster*

Description

Launch a plan of simulation on a cluster

Usage

```
simpleBatchRunSim(
  API_KEY = Sys.getenv("CARBOSEQ_SIWAA_TOKEN"),
  INPUT_DIR,
  chunck_size,
  max_chunck_size = 10000,
  VERBOSE = TRUE
)
```

Arguments

`INPUT_DIR`	the path to the folder where to find the csv input files
`chunck_size`	the size of the chunk of simulations
`VERBOSE`	to get feedback or not

Value

res the result of the plan of simulations

`surveyData` *Survey data*

Description

Loop of refreshing data information, with a 'delay' between 2 requests. Stop as soon as data being in a terminal state or at the end of 'duration_max' (whatever data state).

Usage

```
surveyData(data_id, delay, duration_max, server, api_key, v)
```

Arguments

data_id	Data id
delay	Delay. Unit : second.
duration_max	Duration max. Unit : second.
server	Siwaa server url
api_key	Available Siwaa API KEY
v	Verbose Boolean

Value

data_info Refreshed data information.

---

surveyDatasetCollection

Survey dataset collection

Description

Loop of refreshing dataset collection information, with a 'delay' between 2 requests. Stop as soon as all data (contained into collection) being in a terminal state or at the end of 'duration_max' (whatever data states).

Usage

```
surveyDatasetCollection(
  dataset_collection_info,
  delay,
  duration_max,
  server,
  api_key
)
```

Arguments

dataset_collection_info	Dataset collection information.
delay	Delay. Unit : second.
duration_max	Duration max. Unit : second.
server	Siwaa server url
api_key	Available Siwaa API KEY

Value

dataset_collection_info Refreshed dataset collection information, total_end Boolean (end for all contained datasets), all_data_ready Boolean (all contained datasets ready)

surveyInvocation *Survey workflow invocation*

Description

Loop of refreshing invocation information, with a 'delay' between 2 requests. Stop as soon as invocation being in a terminal state or at the end of 'duration_max' (whatever invocation state).

Usage

```
surveyInvocation(invocation_id, delay, duration_max, server, api_key, v)
```

Arguments

invocation_id	Invocation id
delay	Delay. Unit : second.
duration_max	Duration max. Unit : second.
server	Siwaa server url
api_key	Available Siwaa API KEY
v	Verbose Boolean

Value

invocation_info Refreshed invocation information.

totalEnd *End for all Datasets of a collection*

Description

End for all Datasets of a collection

Usage

```
totalEnd(dataset_collection_info)
```

Arguments

dataset_collection_info	Dataset collection information.

Value

Bolean End for all contained datasets

`tryToGetUserInfo` *Try to get User Info*

Description

Try to get User Info

Usage

```
tryToGetUserInfo()
```

Value

a string

`updateDatasetCollectionInformation` *Update a dataset collection information, adding total_end, all_data_ready*

Description

Update a dataset collection information, adding total_end, all_data_ready

Usage

```
updateDatasetCollectionInformation(env, dataset_collection_info)
```

Arguments

`env`	Siwaa environment
`dataset_collection_info`	Dataset collection information

Value

total_end Boolean, all_data_ready Boolean, refreshed dataset collection information.

uploadAllZips — *Upload a list of .zip input files (from input_dir folder) into a history*

Description

- If 'create_history' is TRUE, then create a history, else work into the existing history defined by 'history_id' (no creation).
- Upload (one by one) the .zip files found under input_dir folder (each .zip file contains its .csv files).

Usage

```
uploadAllZips(env, input_dir, create_history = TRUE, history_name, history_id)
```

Arguments

env	Siwaa environment
input_dir	Path of folder containing the .zip input files.
create_history	Boolean
history_name	Name of history to be created
history_id	Id of existing history

Value

input_data_info_list List of information of uploaded datasets

uploadCSVs — *Upload a set of .csv input files as a .zip file into a history*

Description

1. If 'create_history' is TRUE, then create a history, else work into the existing history defined by 'history_id' (no creation).
2. Upload the .zip file, obtained by zipping 'input_dir' folder that contains the .csv files

Usage

```
uploadCSVs(env, input_dir, create_history = TRUE, history_name, history_id)
```

Arguments

env	Siwaa environment
input_dir	Path of folder containing the .csv files.
create_history	Boolean
history_name	Name of history to be created
history_id	Id of existing history

Value

input_data_info Information of uploaded dataset

message

---

uploadFile *Upload a file into a history*

Description

Note : use the Galaxy tool 'upload1'

Usage

```
uploadFile(file_path, history_id, server, api_key)
```

Arguments

file_path	Path of the file to be uploaded to Siwaa
history_id	History id
server	Siwaa server url
api_key	Available Siwaa API KEY

Value

data_id Id of uploaded dataset

vcat — *Display or not a message depending on 'verbose' or 'silent' mode*

Description

Display or not a message depending on 'verbose' or 'silent' mode

Usage

```
vcat(v, m)
```

Arguments

v	Verbose Boolean
m	message

waitAllSimsResults — *Wait for all simulation results to be ready.*

Description

Loop waiting for a list of datasets to be ready : refresh list of data information, with a 'delay' between 2 requests. Stop as soon as 'total_end' is TRUE or at the end of 'duration_max'.

Usage

```
waitAllSimsResults(env, dataset_collection_info, delay, duration_max)
```

Arguments

env	Siwaa environment
dataset_collection_info	Dataset collection information.
delay	Delay. Unit : second.
duration_max	Duration max. Unit : second.

Details

It is possible to set the survey step between 2 requests ('delay') and the survey total duration ('duration_max').

Possible returned situations at the end of survey :

- 'total_end' TRUE, 'all_data_ready' TRUE : End of process and all data ready to be downloaded.
- 'total_end' TRUE, 'all_data_ready' FALSE : End of process but at least one data not ready (will never be ready).
- 'total_end' FALSE, 'all_data_ready' FALSE : Process still running (not finished). In that case you can call 'waitAllSimsResults' again to continue survey.

Value

dataset_collection_info Refreshed dataset collection information, total_end Boolean (end for all contained datasets), all_data_ready Boolean (all contained datasets ready), data_state_list List of state of collection datasets

---

`waitSimsResults` *Wait for Simulation results to be ready.*

Description

Loop waiting for a dataset to be ready : refresh data information, with a 'delay' between 2 requests. Stop as soon as 'end' is TRUE or at the end of 'duration_max'.

Usage

```
waitSimsResults(env, data_info, delay, duration_max, with_job_report = FALSE)
```

Arguments

`env`	Siwaa environment
`data_info`	Input data information
`delay`	Delay (dataset survey). Unit : second.
`duration_max`	Duration max (dataset survey). Unit : second.
`with_job_report`	Boolean to ask or not for job report

Details

It is possible to set the survey step between 2 requests ('delay') and the survey total duration ('duration_max').

Possible returned situations at the end of survey :

- 'end' TRUE, 'data_ready' TRUE : End of process and data ready to be downloaded.
- 'end' TRUE, 'data_ready' FALSE : End of process but data not ready (will never be ready)
- 'end' FALSE, 'data_ready' FALSE : Process still running (not finished). In that case you can call 'waitSimsResults' again to continue survey.

Value

end Boolean

data_ready Boolean

data_info refreshed data information

job report in case of 'with_job_report' valuing TRUE (job_info information and job_metrics metrics)

message

`waitWorkflowOutputDataIdent` *Wait for the Workflow output data Id being able to be identified, then if that is the case, get and return the output data information.*

Description

Loop waiting for a workflow invocation to reach a terminal state. It refresh invocation information, with a 'delay' between 2 requests. Stop as soon as 'terminal_state' is TRUE or at the end of 'duration_max'. In case of 'scheduled' terminal state, it is possible to access to invocation outputs information. In any other state (other terminal state than 'scheduled' or non terminal state), invocation outputs information are not accessible.

Usage

```
waitWorkflowOutputDataIdent(
  env,
  invocation_id,
  delay,
  duration_max,
  with_more_reports = FALSE,
  v
)
```

Arguments

`env`	Siwaa environment
`invocation_id`	Invocation Id.
`delay`	Delay (invocation survey). Unit : second.
`duration_max`	Duration max (invocation survey) Unit : second.
`with_more_reports`	Boolean to ask or not for more reports
`v`	Boolean to choose 'verbose' or 'silent' mode

Details

It is possible to set the survey step between 2 requests ('delay') and the survey total duration ('duration_max').

Possible returned situations at the end of survey :

- 'terminal_state' TRUE, 'invocation_scheduled' TRUE : Invocation in 'scheduled' terminal state, 'output_data_identified' TRUE.
- 'terminal_state' TRUE, 'invocation_scheduled' FALSE : Invocation in a terminal state other than 'scheduled' (will never be scheduled), 'output_data_identified' FALSE.

- 'terminal_state' FALSE, 'invocation_scheduled' FALSE : Invocation not yet in a terminal state, 'output_data_identified' FALSE. In that case 'waitWorkflowOutputDataIdent' can be called again to continue survey.

To previously launch the workflow invocation : the 'launchWorkflow' method can be used.

Value

output_data_info Output data information, containing output_data_info$history_id, output_data_info$creating_job...

output_data_identified Boolean to know if output_data_info valid

terminal_state Boolean,

invocation_scheduled Boolean,

invocation_info Workflow invocation information

some report summary information (invocation_summary_info and invocation_step_jobs_summary_info)

more reports information in case of 'with_more_reports' valuing TRUE (invocation_steps_info and invocation_report information)

message

Index