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*

Description

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

Usage

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

Arguments

env	Siwaa environment
history_id	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,
  customize_history_name = 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)
customize_history_name	Option to ask for history name customization (default value FALSE : no customization)

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

`defineHistoryName` *Define history name*

Description

History name will be customized or not, depending on customize_history_name. In customization case, complete history name by some user and times information.

Usage

```
defineHistoryName(history_name, customize_history_name)
```

Arguments

`history_name`	History name (to be customized or unchanged)
`customize_history_name`	Boolean to ask for history name customization

Value

historyName Defined history name, customized or not (depending on customize_history_name)

`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 dataset information*

Description

Get information of a dataset

Usage

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

Arguments

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

Details

The dataset information (dataset_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 dataset information (refresh).
- url ('download_url') to download data.
- some data file information ('name', 'file_ext', 'file_size', 'misc_info'...) that may be used when naming the downloaded file.

Done : GET /api/datasets/'data_id' (same as /api/histories/'history_id'/contents/'data_id'

Value

dataset_info Dataset 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 datasetset collection

Get information of a dataset collection of a history

Usage

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

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
data_id	Dataset collection id

Details

The datasetset collection information (dataset_collection_info) contains many fields.

Done : GET /api/dataset_collections/'data_id' (same as /api/histories/'history_id'/contents/'data_id')

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.

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

`getDatasetProvenance` *Get dataset provenance*

Description

Get details related to how dataset was created ('id', 'job_id', 'tool_id', 'stdout', 'stderr', 'parameters', 'inputs' ...)

Usage

```
getDatasetProvenance(
  history_id,
  dataset_id,
  server,
  api_key,
  follow = FALSE,
  brut = FALSE
)
```

Arguments

`history_id`	History id, optional (if missing, get it from data_id)
`dataset_id`	Dataset id Careful : must not be a dataset_collection id ! (ie 'history_content_type' must value 'dataset' into history content, not 'dataset_collection')
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY
`follow`	Boolean to choose to recursively fetch dataset provenance information for all inputs and their inputs, etc.
`brut`	Boolean to force brut format instead of filtered/adapted one

Details

Information filtered (only some fields kept) if not brut

Done : GET /api/histories/'history_id'/contents/'dataset_id'/provenance

Value

dataset_provenance Dataset provenance information.

`getDatasetReports` *Get a set of dataset reports*

Description

Gathering call of some methods according to some boolean activations

Usage

```
getDatasetReports(
  history_id,
  dataset_id,
  server,
  api_key,
  with_dataset_provenance = FALSE,
  follow = FALSE,
  brut = FALSE
)
```

Arguments

`history_id`	History id, optional (if missing, get it from data_id)
`dataset_id`	Dataset id Careful : must not be a dataset_collection id ! (ie 'history_content_type' must value 'dataset' into history content, not 'dataset_collection')
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY
`with_dataset_provenance`	to choose to have dataset_provenance. Default value FALSE.
`follow`	Boolean to choose to recursively fetch dataset provenance information for all inputs and their inputs, etc. Dedicated to with_dataset_provenance case (used only in that case).
`brut`	Boolean to force brut format instead of filtered/adapted one

Value

dataset_provenance Dataset provenance information if with_dataset_provenance.

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.

getInvocationJobs *Get jobs of one invocation*

Description

Giving for each job some information (less than returned by getJobInfo).

Usage

```
getInvocationJobs(invocation_id, server, api_key)
```

Arguments

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

Value

invocation_jobs list of invocation jobs

`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, brut = FALSE)
```

Arguments

invocation_id	Invocation id
server	Siwaa server url
api_key	Available Siwaa API KEY
brut	Boolean to force brut format instead of filtered/adapted one

Details

Information filtered (only some fields kept) if not brut

Done : GET /api/invocations/'invocation_id'/report

Value

invocation_report Invocation report.

getInvocationReports *Get a set of invocation reports*

Description

Gathering call of some methods according to some boolean activations

Usage

```
getInvocationReports(
  invocation_id,
  server,
  api_key,
  with_invocation_steps_info = FALSE,
  with_invocation_report = FALSE,
  brut = FALSE
)
```

Arguments

`invocation_id`	Invocation id
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY
`with_invocation_steps_info`	to choose to have invocation_steps_info. Default value FALSE.
`with_invocation_report`	to choose to have invocation_report. Default value FALSE.
`brut`	Boolean to force brut format instead of filtered/adapted one

Value

invocation_steps_info if with_invocation_steps_info.

invocation_report if with_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, brut = FALSE)
```

Arguments

`invocation_id`	Invocation id
`step_id`	Invocation step id
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY
`brut`	Boolean to force brut format instead of filtered/adapted one

Details

Not filtered (only some fields could be kept)

Done : GET /api/invocations/'invocation_id'/steps/'step_id'

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, brut = FALSE)
```

Arguments

`invocation_id`	Invocation id
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY
`brut`	Boolean to force brut format instead of filtered/adapted one

Details

Not filtered (only some fields could be kept)

Done : GET /api/invocations/'invocation_id'/step_jobs_summary

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, brut = FALSE)
```

Arguments

`invocation_id`	Invocation id
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY
`brut`	Boolean to force brut format instead of filtered/adapted one

Details

Not filtered (only some fields could be kept)

Done : GET /api/invocations/'invocation_id'/jobs_summary

Value

invocation_summary_info Invocation summary information.

`getInvocationSummaryReports`

Get invocation summary reports

Description

Get invocation summary reports

Usage

```
getInvocationSummaryReports(invocation_id, server, api_key, brut = FALSE)
```

Arguments

`invocation_id`	Invocation id
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY
`brut`	Boolean to force brut format instead of filtered/adapted one

Value

invocation_summary_info

invocation_step_jobs_summary_info

getJobCommonProblems *Get job common problems*

Description

Query inputs and jobs for common potential problems that might have resulted in job failure.

Usage

```
getJobCommonProblems(job_id, server, api_key, brut = FALSE)
```

Arguments

job_id	Job id
server	Siwaa server url
api_key	Available Siwaa API KEY
brut	Boolean to force brut format instead of filtered/adapted one

Details

Not filtered (only some fields could be kept)

Done : GET /api/jobs/'job_id'/common_problems

Note : Method only on Galaxy 19.05 or later.

Value

job_common_problems Job common potential problems.

getJobInfo *Get job information*

Description

Note : 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

Done : GET /api/jobs/'job_id'/metrics

Usage

```
getJobMetrics(job_id, server, api_key, brut = FALSE)
```

Arguments

`job_id`	Job id
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY
`brut`	Boolean to force brut format instead of filtered/adapted one

Details

Not filtered (only some fields could be kept)

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)

getJobReports *Get a set of job reports*

Description

Gathering call of some methods according to some boolean activations

Usage

```
getJobReports(
  job_id,
  server,
  api_key,
  with_job_info = FALSE,
  with_job_metrics = FALSE,
  with_job_common_problems = FALSE,
  with_job_resume = FALSE,
  brut = FALSE
)
```

Arguments

job_id	Job id
server	Siwaa server url
api_key	Available Siwaa API KEY
with_job_info	to choose to have job_info. Default value FALSE.
with_job_metrics	to choose to have job_metrics. Default value FALSE.
with_job_common_problems	to choose to have job_common_problems. Default value FALSE.
with_job_resume	to choose to have job_resume. Default value FALSE.
brut	Boolean to force brut format instead of filtered/adapted one

Value

job_info Job information if with_job_info.

job_metrics Job metrics if with_job_metrics.

job_common_problems Job common potential problems if with_job_common_problems.

job_resume Job resume if with_job_resume. Only for job in 'paused' state => Value NULL if job not in 'paused' state.

`getJobResume` *Get job resume*

Description

Resume a job if it is paused.

Usage

```
getJobResume(job_id, server, api_key, brut = FALSE)
```

Arguments

`job_id`	Job id
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY
`brut`	Boolean to force brut format instead of filtered/adapted one

Details

Not filtered (only some fields could be kept)

Done : PUT /api/jobs/'job_id'/resume

Value

job_resume Job resume (list about output dataset associations).

`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

getSomeDatasetReports *Get some reports of a dataset*

Description

Returns some reports of a dataset, depending on boolean activations

Usage

```
getSomeDatasetReports(
  env,
  history_id,
  dataset_id,
  v,
  with_dataset_provenance = FALSE,
  follow = FALSE,
  brut = FALSE
)
```

Arguments

`env`	Siwaa environment
`history_id`	History id (required)
`dataset_id`	Dataset id Careful : must not be a dataset_collection id ! (ie 'history_content_type' must value 'dataset' into history content, not 'dataset_collection')
`v`	Boolean to choose 'verbose' or 'silent' mode
`with_dataset_provenance`	Boolean activator. Default value FALSE.
`follow`	Boolean to choose to recursively fetch dataset provenance information for all inputs and their inputs, etc. Dedicated to with_dataset_provenance case (used only in that case).
`brut`	Boolean to force brut format instead of filtered/adapted one

Value

dataset_provenance Dataset provenance information if with_dataset_provenance.

history_id History id (for memo)

dataset_id Dataset id (for memo)

message

`getSomeInvocationReports` *Get some reports of a workflow invocation*

Description

Returns some reports of a workflow invocation, depending on boolean activations

Usage

```
getSomeInvocationReports(
  env,
  invocation_id,
  v,
  with_summary_invocation_reports = FALSE,
  with_invocation_steps_info = FALSE,
  with_invocation_report = FALSE,
  brut = FALSE
)
```

Arguments

env	Siwaa environment
invocation_id	Invocation Id.
v	Boolean to choose 'verbose' or 'silent' mode
with_summary_invocation_reports	Boolean activator. Default value FALSE.
with_invocation_steps_info	Boolean activator. Default value FALSE.
with_invocation_report	Boolean activator. Default value FALSE.
brut	Boolean to force brut format instead of filtered/adapted one

Value

invocation_summary_info if with_summary_invocation_reports

invocation_step_jobs_summary_info if with_summary_invocation_reports

invocation_steps_info if with_invocation_steps_info

invocation_report if with_invocation_report

invocation_id Invocation Id (for memo)

message

---

getSomeJobReports *Get some reports of a job*

Description

Returns some reports of a job, depending on boolean activations

Usage

```
getSomeJobReports(
  env,
  job_id,
  v,
  with_job_info = FALSE,
  with_job_metrics = FALSE,
  with_job_common_problems = FALSE,
  with_job_resume = FALSE,
  brut = FALSE
)
```

Arguments

`env`	Siwaa environment
`job_id`	Job Id.
`v`	Boolean to choose 'verbose' or 'silent' mode
`with_job_info`	Boolean activator. Default value FALSE.
`with_job_metrics`	Boolean activator. Default value FALSE.
`with_job_common_problems`	Boolean activator. Default value FALSE.
`with_job_resume`	Boolean activator. Default value FALSE.
`brut`	Boolean to force brut format instead of filtered/adapted one

Value

job_info if with_job_info

job_metrics if with_job_metrics

job_common_problems if with_job_common_problems

job_resume if with_job_resume

job_id Job Id (for memo).

message

`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

`isAJobTerminalState` *Define if a job state is or not a terminal state*

Description

Job state : a job is in some non-terminal states then in some terminal states :

- Job terminal states : see JOB_TERMINAL_STATES list
- Job non-terminal states are : 'deleted_new', 'failed', 'new', 'paused', 'queued', 'resubmitted', 'running', 'upload', 'waiting'

Usage

```
isAJobTerminalState(job_state)
```

Arguments

`job_state`	Job state.

Value

Boolean

`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

`isDatasetCollectionInfo` *Define if is data information of a dataset collection*

Description

Criteria : 'history_content_type' value 'dataset_collection' (not 'dataset')

Usage

```
isDatasetCollectionInfo(data_info)
```

Arguments

`data_info`	Data information.

Value

Boolean

`isDatasetInfo` *Define if is data information of a dataset*

Description

Criteria : 'history_content_type' value 'dataset' (not 'dataset_collection')

Usage

```
isDatasetInfo(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

`isJobInATerminalState` *Define if a job is or not in a terminal state*

Description

See isAJobTerminalState for details

Usage

```
isJobInATerminalState(job_info)
```

Arguments

`job_info`	Job information.

Value

Boolean

`isJobOk` *Define if a job is ok or not*

Description

Job ok criteria : 'state':'ok'

Usage

```
isJobOk(job_info)
```

Arguments

`job_info`	Job 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

job_build_for_rerun *Get some information required to rerun a job*

Description

Get details of one job that can be used to rerun the corresponding tool

Usage

```
job_build_for_rerun(job_id, server, api_key)
```

Arguments

job_id	Job Id.
server	Siwaa server url
api_key	Available Siwaa API KEY

Value

Job description with all parameters required to rerun.

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

moreAboutInvocation *Get more information about an invocation (from invocation Id)*

Description

Returns information useful in case of a sequence managed by invocation_id

Usage

```
moreAboutInvocation(
  env,
  invocation_id,
  v,
  with_output_data_info = FALSE,
  with_output_data_report = FALSE
)
```

Arguments

env	Siwaa environment
invocation_id	Id of workflow invocation
v	Boolean to choose 'verbose' or 'silent' mode
with_output_data_info	Boolean to try to get output_data_info
with_output_data_report	Boolean to try to get output data reports

Value

invocation_info Workflow invocation information

history_id Id of history

output_data_identified Boolean to know if output data identified (output data identified only in case of invocation in 'scheduled' state).

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

output_data_info Output data information, Value returned only if with_output_data_info, Value NULL if unidentified output data (ie not output_data_identified)

output_data_reports Output data reports, in case of with_output_data_report (dataset_provenance), Value returned only if with_output_data_report, Value NULL if unidentified output data (ie not output_data_identified)

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).

`relaunchExistingInvocationCarboseqworkflow` *May relaunch existing invocation of 'CarboSeqRun' workflow*

Description

Jobs relaunched in same conditions as initially.

Usage

```
relaunchExistingInvocationCarboseqworkflow(invocation_id, server, api_key)
```

Arguments

`invocation_id`	Id of workflow invocation
`server`	Siwaa server url
`api_key`	Available Siwaa API KEY

Details

Relaunch jobs according to following order/conditions.

Identify the invocation jobs.

Expected : N (unknown) jobs of type CarboSeqSimulator, 1 max job of type CarboSeqSplit, 1 max job of type CarboSeqUnsplit.

Study the job of type CarboSeqSplit, then together jobs of type CarboSeqSimulator, then the job of type CarboSeqUnsplit.

If job of type CarboSeqSplit relaunched, then no job relaunched in types CarboSeqSimulator, CarboSeqUnsplit. If at least one job of type CarboSeqSimulator relaunched, then job of type CarboSeqUnsplit not relaunched.

A job may be relaunched if in a terminal state other than 'ok'.

Value

relaunch_refused Boolean Not in situation to relaunch. For example more than one job of type CarboSeqSplit, more than one job of type CarboSeqUnsplit, some job not being remappable...

job_relaunched_ids Ids list of relaunched jobs.

job_ids Ids list of all jobs

job_states States of each job

job_remaps job_remap value of each job for which job_build_for_rerun has been done

job_CarboSeqSplit_ids

job_CarboSeqSimulator_ids

job_CarboSeqUnsplit_ids

---

rerunJob *Rerun a job*

Description

Job rerun in same conditions as initial run : tool parameters, history.

Usage

```
rerunJob(job_id, server, api_key)
```

Arguments

job_id	Job Id.
server	Siwaa server url
api_key	Available Siwaa API KEY

Details

The job output(s) will be remapped onto the dataset(s) created by the original job (cf job_inputs$rerun_remap_job_id).

If other jobs were waiting for this job to finish successfully, they will be resumed using the new outputs of this tool run.

Value

ret valuing NULL if no rerun (case if job is not remappable, cf job_remap) and else containing Information about outputs and rerun job.

job_remap value

---

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, output_filename)
```

Arguments

env	Siwaa environment
data_info	Data information
output_dir	Path of folder destination of downloaded file
output_filename	Name of downloaded file (containing the extension)

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,
  customize_history_name = FALSE
)
```

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.
customize_history_name	Option to ask for history name customization (default value FALSE : no customization)

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

API_KEY	param
INPUT_DIR	the path to the folder where to find the csv input files
chunck_size	the size of the chunk of simulations
max_chunck_size	param
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

---

tryRelaunchWorkflow *Try to relaunch an existing Workflow invocation*

Description

Identify the invocation jobs, try to relaunch some of them (being in a terminal state other than 'ok') and return some information describing what has been done.

Usage

```
tryRelaunchWorkflow(env, invocation_id, v)
```

Arguments

`env`	Siwaa environment
`invocation_id`	Invocation Id.
`v`	Boolean to choose 'verbose' or 'silent' mode.

Value

relaunch_refused Boolean Not in situation to relaunch. For example more than one job of type CarboSeqSplit, more than one job of type CarboSeqUnsplit, some job not being remappable...

job_relaunched_ids Ids list of relaunched jobs.

job_ids Ids list of all jobs

job_states States of each job

job_remaps job_remap value of each job for which job_build_for_rerun has been done

job_CarboSeqSplit_ids

job_CarboSeqSimulator_ids

job_CarboSeqUnsplit_ids

message describing what has been done.

`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 (dataset) 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,
  v,
  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.
v	Boolean to choose 'verbose' or 'silent' mode
with_job_report	Boolean to ask or not for job reports.

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

output_data_reports Output data reports, always returned (dataset_provenance).

job_reports Some job reports in case of with_job_report (job_info, job_metrics, job_common_problems, job_resume).

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_invocation_report = 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_invocation_report	Boolean to ask or not for more invocation 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

summary_invocation_reports Some invocation reports summary, always returned (invocation_summary_info, invocation_step_jobs_summary_info).

output_data_reports Output data reports, always returned (dataset_provenance). Value NULL if not output_data_identified.

invocation_reports Some invocation reports, returned in case of with_more_invocation_report (invocation_steps_info, invocation_report).

message

Index