



Talend User Component tRunTask

Purpose

This component runs job as task in the Talend Administration Center (TAC).

The advantages are:

- It is possible to create execution plans with nearly unlimited flexibility
- It enables the chaining of external job scheduler with the TAC
- It is not necessary to build “monster” jobs by embedding the whole process with tRunJob
- With chaining task with tRunTask it is always possible to monitor all single steps (one task)
- The used API does not need Talend specific libraries and is based on Open Source technology
- Since Talend release 5.6.1 the component returns the exit code from the job as return code.
- Has two modes:
 - Run a Task: Runs a task
 - Retrieve a task status: This mode reads the current status of a task and provide it as return values

Talend-Integration

This component can be found in the palette under Management (next to tRunJob)

This component provides several return values.

Basic settings

Property	Content
Operational Mode	Run Task: Run a task Check t
TAC URL	URL of the TAC (it is the same as used in Studio or in the browser) required
TAC login	User login (User need the Administrator role) It is recommended to use an technical user
TAC password	Users password
Use task label	If false you need to know the ID of the task (refer to the information view of a task in the TAC)
Task label is job name	In case of the task has as label the job name check this option. In this case the job name is used as task label.
Task ID	ID of the task required (if Use task label is false)
Task label	Label of the task required (if Use task label is true)
Job run by task	The job, which will be run by this task. This configuration expects to choose the job which is actually used in the already deployed task. It does NOT change the job for a task. The purpose is to configure the context variables.

Additional basic settings for mode: Run a task

Property	Content
Context Parameters	The context parameters and its source.
Run task asynchronously	If true it starts the task and pools for its end. If false it waits for the end in the same http request (in long running task this could lead to broken pipes)
Wait until the end	Waits for the end of the job. Otherwise the job will be started and the component finish and the current job can continue. It is like fire and forget. To prevent calling a task twice at the same time, the component checks at if the task is already running and waits until its end.
Check time cycle until job is running	If Run task asynchronously is true, we have to wait until the TAC has started the job the poll on its end. Set the time in ms.

Check time cycle until job is running	If Run task asynchronously is true, we have to poll on its end. It depends on the experiences about the typical task run duration. It is not recommended to pool too often, because it could lead to a notable load for the TAC.
Wait until this task and none of the task in the list are running	If true, the component pools on the finish status of its own task and the listed tasks and starts its own task after none of these tasks are currently running.
Die on error	Dies if the started task fails (means the job fails because of any problems within the job). If there are any errors in communication with the TAC, the tRunTask component will always fail.

Advanced settings

Property	Content
Debug output requests and response	If true the component prints out the requests and the responses to standard output stream
Allow task generating or deploying	if false: the component fails if the task is not in the status: Read To Run. if true: the component starts the job and checks the preparing status. This will also allow the task to be generated between two runs.
Maximum repetition in case of TAC errors	If the TAC returns errors the component repeats the request multiple times. If the maximum number of repetition is reached, the component fails.
Wait time between repetition	Wait time in milliseconds after a TAC error happened before the next attempt.

Return values

Return value	Content
ERROR_MESSAGE	Last error message. Unfortunately this is not the error message from the actually running job. This message is built from the tRunTask component. The current TAC web service does not provide this message.
TASK_ID	The task id retrieved from the TAC by the task label. All commands to the TAC related to task need the task ID. That's why the tRunTask component retrieves this ID at first.
RUN_DURATION	The time the task is running measured by the tRunTask component. It is not exactly the time because of the possible polling time delay.
RETURN_CODE	Before Talend release 5.6 the component is not able to get the real job exit code and returns 4 as exit code in case of the task run fails (real job failures). Starting with Talend release 5.6 the component delivers the real exit code of the job. This real exit code is only available in the "run" mode! In the status mode the component has no access to the real exit code and returns 4 in case of errors and 0 in case of OK.
HAS_ERRORS	The job has errors (the last run of the job (task) failed)
IS_READY_TO_RUN	The job is ready to run
IS_PREPARING	The job needs preparing like generating or deploying
IS_RUNNING	The job is currently running
STATUS	The current status of the job as text
ERROR_STATUS	The current error status of the job

Scenario 1:

Simply running a task:

The screenshot displays the Talend Designer interface for configuring a job. The job is named 'Job(test_runtask 0.1.r613)' and is currently in the 'Designer' view. The job flow consists of a 'tFileList_1' component connected to a 'load stage' component via an 'Iterate' connector. The 'load stage' component is configured with the following settings:

- Basic settings:**
 - TAC URL: `http://on-0337-jll.local:8080/org.talend.administrator.511`
 - User: `jan.lolling@cimt-ag.de`
 - Password: `loli`
 - ☒ Use task label (retrieve the task-ID by label)
 - Task label (Label of the task in the TAC, this is NOT the job name!): `test1`
 - Job run by task (only to get the context here!): `test_job`
 - Version: `Latest`
 - Context: `Default`
- Context Parameters:**

Parameter	Value
param_int	100
param_double	56.789d
param_date	TalendDate.getCurrentDate()
param_string	((String)globalMap.get("tFileList_1_CURRENT_FILEPATH"))
param_boolean	true
die_code	0
- Run task asynchronously:** ☒ **Wait until task ends:** ☒
- Check time cycle until job is running:** `200` **Timeout for check until running:** `10000` **Check time cycle until job has been finished:** `1000`
- Wait until this task and none of the task in the next list are running:** ☒
- List of task:**

Task name

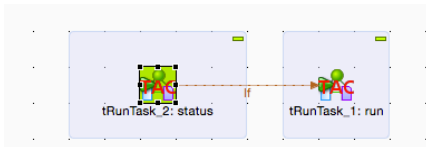
- Die on error (errors in task, errors while requesting TAC will always stop processing):** ☒

This scenario shows the way to implement a trigger, which starts a task in the TAC for every found file. There are a lot of other scenarios possible.

One of the most used scenarios is to trigger a task from another Job scheduler because of a company policy about scheduling. In large companies there are typically dedicated schedulers and with this component you can write a job with it self can started as simple script from such kind of schedulers.

Scenario 2: Watchdog job

In this scenario a job check the status of a task and if the task has been failed the task will be run again.
In a real world scenario such job will be run also as task in the TAC and would be triggered several times within a hour.



This is the simple job design.

Here the basic settings for the tRunTask_2. The enhanced label can be achieved by set as label in the View setting:
__UNIQUE_NAME__: __MODE__

tRunTask_2: status(tRunTask_2)

Basic settings Operational Mode: Retrieve a task status ▼*

TAC Connection

TAC URL: context.tac_url

User: context.tac_user * Password: context.tac_passwd

☒ Use task label (retrieve the task-ID by label) ☒ Task label is job name

Job run by task (only to get the context here!): test_truntask_testjob ... Version: Latest ▼* Context: Default ▼*

The if condition is: ((Boolean)globalMap.get("tRunTask_2_HAS_ERRORS"))
This is a return value and can be put here by drag and drop.

If the task has errors, the task should be run again:
Here the basic settings of the tRunTask_1:

tRunTask_1: run(tRunTask_1)

Basic settings Operational Mode: Run a task ▼*

TAC Connection

TAC URL: context.tac_url *

User: context.tac_user * Password: context.tac_passwd *

☒ Use task label (retrieve the task-ID by label) ☒ Task label is job name

Job run by task (only to get the context here!): test_truntask_testjob ... Version: Latest ▼* Context: Default ▼*

Context Parameters

Parameter	Value
die_code	0
boolean_value	true
date_value	TalendDate.getCurrentDate()
string_value	"uäö\\nXX"
integer_value	((Integer)globalMap.get("tLoop_1_CURRENT_ITERATION"))
bigdecimal_value	new java.math.BigDecimal("1.23456789")

☒ Run task asynchronously ☒ Wait until task ends

Check time cycle until job is running: 200 * Timeout for check until running: 10000 * Check time cycle until job has been finished: 1000 *

☒ Wait until this task and none of the task in the next list are running

List of task

Task name

☐ Die on error (communication errors to the TAC will always let the job die)