

TASKWORKER

CBM Archives Company

Document Version 1.1

Last modified on 12/13/2013

Contents

[Introduction 3](#_Toc371665141)

[System Requirements 3](#_Toc371665142)

[Setup & Configurations 3](#_Toc371665143)

[Setting the running options 3](#_Toc371665144)

[Option 18: Logging format 5](#_Toc371665145)

# Introduction

Taskworker is a platform for running background processes that are developed using the Microsoft .Net framework. Company growth brings challenges for computing infrastructure and application services. Often many systems are brought onboard to fulfill business requirements but are loosely connected and become difficult to manage. Taskworker provides a means to centralize business processes that can be managed and scalable.

# System Requirements

* Intel® Pentium® 4 or AMD Athlon® 64 processor (2GHz or faster)
* Windows 7 with Service Pack 1, Windows 8, or Windows 8.1
* 1GB of RAM

# Setup & Configurations

### Setting the running options

After running the application installer you should have a desktop icon. Double click it to get started.

The title screen will appear along a list of options 1 through 19.

Option 1: Run As Service

Set this option to false to allow the application to run under a Windows console. This option must be set to true when running as a windows service.

Option 2: Show Form

This option enables any console window or windows form(s) to appear during operation.

Option 3: AppConnectionStr

Mandatory Option: Application connection must point to the database that contains the taskworker tables. If the tables do not exist you will be asked if you want the created.

Option 4: procRunName

Mandatory Option: This is the stored procedure that returns pending jobs for processing.

Option 5: initstr

Not in use.

Option 6: maxtasks

Set the maximum tasks to be performed

Option 7: appCode

Future option

Option 8: appversion

Future option

Option 9: appSection

Future option

Option 10: LoadDBSettings

This part of a future use feature. Keep this value to false.

Option 11: Respawn in secs

Time to wait before starting up

Option 12: Action Time To Live (sec)

How long to let a single process live. If a process hangs this option will kill if it exceeds the action time to live.

Option 13: Wait for each thread to end

Setting to true makes Taskworker wait for a process to end before starting a new one. Setting to false will launch each queued process at the same time.

Option 14: Max Error Count

How many times a process can be re-tried before it is skipped by taskworker

Option 15: Logs

Path that will contain Taskworker logging files.

Option 16: Log to file

Turns logging by file on/off

Option 17: Log values [separate by commas]

Each event has a numerical value. Values to report on must be separated by commas.

1 - Successful

2 - Warning

3 - Error

20 - Info level 1

30 - Info level 2

101 – debug 1

102 – debug 2

103 – debug 3

104 – debug 4

Option 18: Logging format

Logging options are HTML or TEXT

Option 19: Logging CSS path

If HTML logging format is selected a style sheet can be used to enhance the marked up returned in the log file. Specify the full or relative path to a .css file.

Option Q - Quit

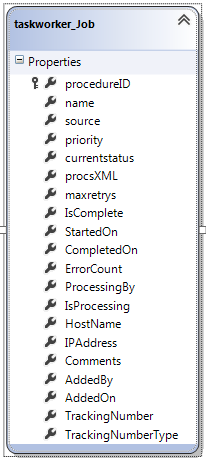
Quits the application

Option R - Run

Runs Taskworker

## Jobs Table

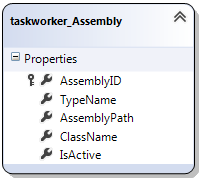
TaskWorker depends SQL server returning XML structures that contain job information. Typically this involves a stored procedure returning rows from a job table [taskworker\_jobs]. For use with the Management Console this table contains the following fields:



TaskWorker calls on the store procedure defined on option #4 to select pending records from the taskworker\_jobs table. [procsXML] is one of the few fields returned. This field contains job information.

## Actions

Inside the job XML structure are [action] elements. Actions are defined in the [taskworker\_Assembly] table. Actions may be located in a single assembly or several.



Example action XML element:

<connectionstrings>

<connection name="db1" value="Data Source=SRV1;Initial Catalog=XDA01;User ID=User\_01;"/>

<connection name="db2" value="Data Source=SRV2;Initial Catalog=XXA01;User ID=User\_02;”/>

</connectionstrings>

<parameters>

<parameter name="expire\_minutes" value="120" />

<parameter name="path\_ToBeProcessed" value="\\srv1\vol1\pending\_files" />

<parameter name="path\_Processed" value="\\srv1\vol1\Processed" />

<parameter name="path\_UnableToProcess" value="\\srv1\vol1\UnableToProcess" />

<parameter name="path\_Logs" value="\\srv1\vol1\Logs" />

<parameter name="path\_outfiles" value="\\srv1\vol1\Export" />

<parameter name="process\_sp\_name" value="abc\_email\_notify" />

<parameter name="file\_startswith" value="INV" />

<parameter name="file\_regex" value="" />

</parameters>

</action>

## Troubleshooting

Problem: Taskworker is running but appears to have nothing to process. Files are stacking up and there are no apparent issues or error messages.

Resolution: SQL Agent is not running (started). Check this service and start it. Also, check the jobs to see if they are enabled.

Problem: Receiving “workflow automation” email notifications. TaskWorker has either crashed or is no longer running.

Resolution: Remote into the SQL Server as administrator and ensure Taskworker is running.

Problem: Taskworker is running but there is a message about SQL connection timing out.

Resolution: Verify connectivity to the SQL server and SQL service is running.

Note: If an error or issue persists please contact DLX for support.

# RobRoy Configuration

## Option Values

1. RUNASSERVICE? (False)

2. show form? (True)

3. appConnectionStr (Data Source=dlogsql;Initial Catalog=DocumentManager;Persist Security Info=True;User ID=dmclient; password=dmclient;)

4. procRunName (dlx\_RR\_CHESS\_getjobs)

5. initstr ()

6. maxtasks (3)

7. appCode (taskworker)

8. appversion (1.0.0.1)

9. appSection (dlx\_rr\_worker01)

10. LoadDBSettings? (False)

11. Respawn in secs (60)

12. Action Time To Live (sec) (500)

13. Wait for each thread to end? (True)

14. Max Error Count (5)

15. Logs directory (f:\TaskworkerLogs)

16. Log to file? (True)

17. Log values [separate by commas] (1,2,20,30,101,102,103,104)

18. Logging format (HTML)

19. Logging CSS path (..\Styles\logFiles.css)

## Job Definitions

All RR jobs are created and scheduled using SQL Server Agent. To make any changes you must connected to DLOGSQL using SQL Server Management Studio. The following are the current job definitions:

RR\_DM\_HR\_TASKS

RR\_DM\_PROCESSES

RR\_DM\_REPORTS

RR\_DM\_RETENTION

RR\_DM\_SPECIAL\_PRICING

Each job runs on its own schedule and contains SQL scripting for creating job(s). Edit a job step to make changes to the script. For example, the script for the UlitiPro file looks like the following:

declare @xdoc as xml

set @xdoc =

N'<procedures initialstatus="pending" status\_error="error\_aborted" status\_onretry="error\_willretry" status\_onsuccess="all\_completed">

<procedure type="HR\_TASKS" runorder="1" status="pending">

<actions>

<action type="process\_UltiPro\_Import" description="process\_UltiPro\_Import" runorder="1" status="pending">

<connectionstrings>

<connection name="rrdm" value="connection timeout=240; Data Source=dlogsql;Initial Catalog=documentmanager;Persist Security Info=True;User ID=dmclient; Password=dmclient; Connection Timeout=400;" />

<connection name="chess" value="connection timeout=240; Data Source=dlogsql;Initial Catalog=documentmanager;Persist Security Info=True;User ID=dmclient; Password=dmclient; Connection Timeout=400;" />

</connectionstrings>

<parameters>

<parameter name="path\_ToBeProcessed" value="\\doclog\apexport$\Sorting Office\HR-Files-Pending\" />

<parameter name="path\_Processed" value="\\doclog\apexport$\Sorting Office\Processed" />

<parameter name="path\_UnableToProcess" value="\\doclog\apexport$\Sorting Office\UnableToProcess" />

<parameter name="path\_Logs" value="\\doclog\apexport$\Sorting Office\Logs" />

<parameter name="path\_outfiles" value="\\doclog\apexport$\Chess Export\Reports" />

<parameter name="path\_tmp" value="\\doclog\apexport$\Sorting Office\tmp" />

<parameter name="hr\_filename" value="Recruitment\_Data\_Project.xml" />

<parameter name="process\_sp\_name" value="ews\_CreateRecords" />

</parameters>

</action>

</actions>

</procedure>

</procedures>'

insert into TWK\_RunProcedures

(name,source, priority, currentstatus, procsXML, maxretrys, IsComplete, AddedBy, AddedOn)

values ('process\_UltiPro\_Import','auto',1,'pending',@xdoc, 10, 0, 'DLX-SYSTEM', GETDATE())

## Changes can be made to any of the parameters and saved for future runs.