

IccParseTaskScheduler Manual

Contents

Description.....	1
IccParseTaskScheduler	1
Installation.....	2
Logic File Description/Syntax.....	2
Logic File.....	2
Filter Types	7
Example Logic File.....	8
Definitions	9
Modifications.....	9

Description

This document describes how to use the IccParseTaskscheduler.

IccParseTaskScheduler

is a Command Line program created to query/report back on the Tasks running on machines. The program runs a thread for each machine provided, i.e. querying all at the same time.

We recommend using IccInvestigateRecords for displaying the results in a web/searchable tool, ref:

<http://services4.lowercolumbia.edu/demo/projectlid/IccInvestigateRecords>

Installation

- copy the lccParseTaskScheduler.exe to a folder
- create a Logic File
- run the lccParseTaskScheduler.exe lcc:logicPath [...logic file...]

Logic File Description/Syntax

A Logic File is a Tab delimited text file. Any lines not recognized as a valid Key/Value pair, will be ignore and can be used as remarks/other.

The Logic File uses the syntax.

Syntax: **[Key]** [tab] **[Value]** ... [tab] **[Value]**

Example: lcc:key value

Any extra tabs in a line after the expected ones are considered remarks and will be ignored. This is a nice way to document specific Key settings (see Log Levels in the Logic File example(s) for reference). Also, if you place a tab before a line, that will essentially make it a remark and will be ignored, which makes using/not using logic without removing quicker.

Any line not starting with an expected key is ignored, which makes placing remarks/formatting the logic easy.

Logic File

lcc:logPath *(mandatory, one to many per Logic File)*

Supported modules: **Front End, Back End, Admin Decryption**

The path/root name of the Log File. The program will place the '.log' extension automatically and will append a Year/Month date.

Syntax: lcc:logPath [tab] [..Path..]

Example: lcc:logPath e:\ourLogs\lccROAR\lccROAR

lcc:debugLevel *(optional)(none to many per Logic File)*

Specifies what debugging changes will take effect. Provide a new line for each level desired.

Debug Levels

- 1 Logic Settings: show what Logic Settings are loaded
- 2 Process Flags: show what happens to value as flags are applied
- 3 Do Not Clean Up Thread OutputFiles: as threads are executed against each machine, the output is recorded to temp files to be read from later when creating the reports. These temp files are automatically cleaned out after done, unless this Debug Level is provided.

lcc:machine *(mandatory, one to many per Logic File)*

What machine to query.

Syntax: `lcc:machine [tab] [..machine name..]`

Example: `lcc:machine ourServer1`

Example #2 (multiple):

`lcc:machine ourServer1`

`lcc:machine ourServer2`

lcc:validTaskStatus *(optional, one to many per Logic File)*

If not provided, all tasks are valid to read.

Normally 'Ready' is provided.

Syntax: `lcc:validTaskStatus [tab] [...]`

Example: `lcc:validTaskStats Ready`

Example #2 (multiple):

`lcc:validTaskStats Ready`

`lcc:validTaskStats Disabled`

lcc:filterSetId *(optional, one to many per Logic File)*

Starts a new Filter Set. The 'Id' is just for logging and does not affect processing.

Syntax: `lcc:filterSetId [tab] [..Id..]`

Example: `lcc:filterSetId our set 1`

lcc:filterSetHostName *(optional, one to many per Logic File/Filter Set)*

Provides filter(s) for Host Names. See Filter Types section below.

This will look at the Host Name column of each task.

Valid Types:

- Equals
- Does Not Equal
- Contains
- Does Not Contain

Syntax: **lcc:filterSetHostName** [tab] [Type] [tab] [..Id..]

Example: **lcc:filterSetHostName** Equals ourMachine

lcc:filterSetTaskName *(optional, one to many per Logic File/Filter Set)*

Provides filter(s) for Task Names. See Filter Types section below.

This will look at the Task Name column of each task.

Valid Types:

- Equals
- Does Not Equal
- Contains
- Does Not Contain

Syntax: **lcc:filterSetTaskName** [tab] [Type] [tab] [..Id..]

Example: **lcc:filterSetTaskName** Equals ourTask

lcc:filterSetTaskToRun *(optional, one to many per Logic File/Filter Set)*

Provides filter(s) for Task To Run. See Filter Types section below.

This will look at the Task To Run column of each task.

Valid Types:

- Equals

- Does Not Equal
- Contains
- Does Not Contain

Syntax: `lcc:filterSetTaskToRun [tab] [Type] [tab] [...Id..]`

Example: `lcc:filterSetTaskToRun Equals ourTask`

lcc:filterSetLastResult *(optional, one to many per Logic File/Filter Set)*
Provides filter(s) for Last Result. See Filter Types section below.

This will look at the Last Result column of each task.

Valid Types:

- Equals
- Does Not Equal
- Contains
- Does Not Contain

Syntax: `lcc:filterSetLastResult [tab] [Type] [tab] [...Id..]`

Example: `lcc:filterSetLastResult Equals ourTask`

lcc:SMTPServer *(optional)(one per Logic File)*
Email server to send email alerts to.

Syntax: `lcc:SMTPServer [tab] [Name]`

Example: `lcc:SMTPServer ourEmailServer.edu`

lcc:SMTPPort *(optional)(one per Logic File)*
Port to the SMTP server .

Syntax: `lcc:SMTPPort [tab] [#]`

Example: `lcc:SMTPPort 25`

lcc:SMTPSSL *(optional)(one per Logic File)*
Forces the program to use SSL with the email server.

Syntax: `lcc:SMTPPort [tab] [#]`

Example: `lcc:SMTPPort 25`

lcc:SMTPSubject *(mandatory)(one per Logic File)*

Provides the email subject.

Syntax: `lcc:SMTPSubject [tab] [Subject]`

Example: `lcc:SMTPSubject lccSQLTableChecker Warning`

lcc:SMTPBodyPath *(mandatory)(one per Logic File)*

Provides the email body content template to use when emailing.

See section SMTP Body Template section on flags that can be used.

Syntax: `lcc:SMTPBodyPath [tab] [...path...]`

Example: `lcc:SMTPBodyPath .\HTMLs\HTML-SMTP.htm`

lcc:SMTPSender *(mandatory)(one per Logic File)*

Provides the from email address and name that messages are addressed from.

Syntax: `lcc:SMTPSender [tab] [Address] [tab] [Name]`

Example: `lcc:SMTPSender ourmailbox@ourcollege.edu Our Mailbox`

lcc:SMTPUserId *(optional)(one per Logic File)*

If email server doesn't allow Anonymous authentication, this Key provides the User credentials id.

Syntax: `lcc:SMTPUserId [tab] [Id]`

Example: `lcc:SMTPUserId ouruser`

lcc:SMTPUserPassword *(optional)(one per Logic File)*

If email server doesn't allow Anonymous authentication, this Key provides the user credentials Password.

We highly recommend using the Encrypting Values section/option.

Syntax: `lcc:SMTPUserPassword [tab] [Password]`

Example: `lcc:SMTPUserPassword` our user password

lcc:SMTPRecipient (*mandatory*)(*one to many per Logic File*)

What recipient should receive an email alert. Provide this for each recipient.

Syntax: `lcc:SMTPRecipient [tab] [Address] [tab] [Name]`

Example: `lcc:SMTPRecipient ourteam@college.edu Our Team`

Example: `lcc:SMTPRecipient specificperson@site.com Specific Person`

lcc:reportPath (*mandatory, one per Logic File*)

What file path to write the report to.

Syntax: `lcc:reportPath [tab] [..path..]`

Example: `lcc:reportPath .\folder\ourReport.txt`

SMTP Body Template

The key "lcc:SMTPBodyPath" will load the content in that path to be used as the email template for sending email alerts.

The content should be a full web page (HTML), with all core HTML code needed.

The page can also contain 'flags' that will auto replace with content when sending the email.

Flags:

- [lcc:subject] : the subject
- [lcc:senderName] : the sender's name
- [lcc:senderAddress] : the sender's address
- [lcc:recipientName] : the recipient's name
- [lcc:recipientAddress] : the recipients address

Filter Types

Filter Types control how Filters are matched.

TYPES

- Equals : a value must equal to match. Any of the filters in a set is considered a match for the entire list.
- Does Not Equal : a value must not equal to match. A value must not equal any in a set.
- Contains : a value must contain to match. Any of the filters in a set is considered a match for the entire list.
- Does Not Contain : a value must not contain to match. A value must not contain any in a set.

Flag Replace

The Logic File can have flags embedded into values and have those replaced on the fly when the program is ran.

To embed a Flag Replace:

Syntax: `[lccFlagReplace:...FLAG...]`

Example: `[lccFlagReplace:YYYYMMDD]`

Flags Available:

- **YYYYMMDD** : replaces with Year Month Day
- **YYYYMMDDHHMMSSMS**: replaces with Year Month Day Hour Minute Second Millisecond
- **HHMMSS** : replaces with Hour Minute Second
- **HHMMSSMS** : replaces with Hour Minute Second Millisecond
- **HH:MM:SS** : replaces with Hour : Minute : Second
- **HH:MM:SS.MS** : replaces with Hour : Minute : Second : Millisecond

Example:

`lcc:reportPath.\reports\lccParseTaskScheduler-report-alerts-[lccFlagReplace:YYYYMMDD].csv`

Example Logic File

```
lcc:debugLevel      1      logic settings
lcc:debugLevel      2      process flags
lcc:logPath         .\lccParseTaskSchedulerLog

lcc:machine         ourServer1
```

```
lcc:machine ourServer2
lcc:machine ourServer3
lcc:reportPath.\lccParseTaskScheduler-report.txt

lcc:filterSetId 1
    lcc:filterSetSkip YES
lcc:filterSetHostName Equals ourComputer
lcc:filterSetTaskName Equals \ourTask
lcc:filterSetTaskToRun Equals C:\directory\file.bat
lcc:filterSetLastResultDoes Not Equal 0

lcc:filterSetId 2
    lcc:filterSetSkip YES
lcc:filterSetHostName Equals ourComputer2
lcc:filterSetTaskName Equals \ourTask2
lcc:filterSetTaskToRun Equals M:\directory\directory\script.bat
lcc:filterSetLastResult Equal 0
```

Definitions

Task – a task configured to run through Task Scheduler

Modifications

NAME	DATE	MODIFICATION
David Mielcarek	20180312	Created
David Mielcarek	20250721	Added Debug Level '3'. Added Keys 'lcc:validTaskStatus', 'lcc:filterSetId', 'lcc:filterSetHostName', 'lcc:filterSetTaskName', 'lcc:filterSetTaskToRun', 'lcc:filterSetLastResult', 'lcc:SMTP...' (keys)
David Mielcarek	20250805	Added lccFlagReplace section.

End of document