Corticon Logging

Corticon logging provides fast and flexible logging functionality for production Servers and Studio tests.

Beginning in Corticon 5.5 we provide greater control over the information that is written to your logs. Using the brms.properties file you can control where log files are created, whether to rollover the logs, how files are retained, what type of information is written to those files and even separate Decision Services into separate log files.

The brms.properties file can be used to override the default log file management.

- You can control where the logs are written using the logpath. The default value is your work directory, but you can uncomment the line and specify a directory of your choice. It is worth noting that you must use forward slashes as a path separator as shown here.
- The logDailyRollover specifies whether to rollover the logs based on elapsed time and size. Default value is true.

You also have control over how many logs to retain using the logRolloverMaxHistory. The default value is 5.

Two brms.properties control what is written to the logs. The loglevel controls What level of logging is used. Values range from OFF to ALL. The default is INFO, shown here. At INFO and higher levels, you can add details in the logs using the logFiltersAccept. The default messages that are added are DIAGNOSTIC and SYSTEM, Using a comma separated list you can pick and choose just the entry types you want logged. Setting it equal to blank does not add any of the messages.

Let's take a closer look at the log levels. The lowest level is OFF which creates a log file with version details only. The default loglevel is INFO providing basic information. From INFO to ALL you can selectively filter the information by using the logFiltersAccept.

The logFiltersAccept lets you include different types of information emanating from running services to the log. When the loglevel is set to INFO or higher, this property accepts logging of information types that are listed. Set your preferred log filters by uncommenting the line # logFiltersAccept= in brms.properties, and then listing the functions you want to have in the logs as a comma-separated values from the following:

RULETRACE records complete content of each request and response.

DIAGNOSTIC records server and Decision Service performance diagnostics at a defined interval (default is 30 seconds).

TIMING records timing events.

INVOCATION records invocation events

VIOLATION records exceptions

INTERNAL records internal debug events

SYSTEM records low-level errors and fatal events

Using these shortcuts we'll generate log activity.

Always save a pristine copy of the brms.properties before making edits.

In this test, we open a file with the defaults settings and single change to the logpath.

The top of the file contains an introduction to the new log settings.

Beneath the initial comments are a series of comments that show the default settings.

We've kept those as a reminder in case we want to reset them to the defaults.

They only change made here was to store the log file in a directory called DEFAULT so we can separate our test cases.

Now we save as brms.properties overwriting the existing file

Next step is starting the server. Waiting for a server startup message to appear.

Step 3 is launching the testing script testServerAxis.bat that is shipped with the product.

Enter 131 to send a test file to the processOrder Decision Service.

Enter the default file name for the request document to submit a valid request.

Enter 131 again and supply 'badfile' to generate an error message. Then close the test window.

Next step is to stop the server. This stops the logging and prepares it for the next test.

Now we can review the log file.

We navigate to the logpath directory for this test called DEFAULT and open CcServer.log. The log file is only 9K. You'll want to remember that number for later.

Searching for INFO reveals the types of message captured at the log level.

Next search for INFO SYSTEM to see the types of messages added by the LogFiltersAccept include SYSTEM messages.

Next we search for INFO DIAGNOSTIC to locate the messages written to our log.

There is a second log created by the use of the testServerAxis.bat called the Corticon.log. Opening this file shows the ERROR messages that were generated by the badfile name.

Now let's turn all logging off using brms_NONE.properties file.

The loglevel equal OFF and the logFiltersAccept is equal to blank.

Save as the brms.properties.

I've repeated steps 2-4. And we return to Step 5 to review the logs.

Opening the CcServer.log shows only the license details.

Now we'll enable all the logging. This would typically be done only at the request of a Technical Support Engineer to solve a particular problem and then reset to the defaults.

Loglevel is ALL. All filters are specified in a comma separated list.

And we save as brms.properties.

We fast forward to review the results. The first thing to note is the size of the file. The default file was 9K and this file is 350K.

We have INFO and DEBUG level messages that show the cumulative nature of the loglevel.

Within the DEBUG messages we have the INTERNAL messages appearing we added to the logFiltersAccept.

Searching for RULETRACE provides more details about the INPUTS that were sent during the test.

The Corticon.log has grown from 1K to 13K.

In special cases you may want to temporarily separate the log at the Decision Service level.

Setting the logPerDS property equal to true turns on decision service logging.

The steps are the same.

For more information on different use case please see the following topics in the documentation. This concludes our look at Corticon logging.