Genesys Web Engagement

log Section

4/30/2025

log Section

affectedLoggers

Default Value: The default value is an empty string, which means that there aren't any affected loggers.
Valid Values: Comma-separated list of logger names, specified in LOG4J2.xml. For example: com.genesyslab.webme.commons,PROTOCOL,org.apache.cassandra
Changes Take Effect: Immediately

Verbosity settings are explicitly applied for some loggers, as discussed in the extended description.

Extended description

Verbosity settings are explicitly applied for the following loggers:

- Loggers that are not declared explicitly in the `log4j2.xml` configuration file.
- Loggers that are specified explicitly in the `log4j2.xml` and are specified in the value for this `affectedLoggers` option.

For other loggers specified in `log4j2.xml`, but not mentioned in the value for this option, the verbosity level is not re-applied.

Here is a use case for when you might need to set this option:

- Cassandra needs to write error messages to a log file, and at the same time, Genesys components also need to write debug messages to the log file.

To resolve this use case, you would:

1. Specify the following logger in `log4j2.xml`: `<logger name="org.apache.cassandra" level="error" additivity="false">`
2. **Do not** include `org.apache.cassandra` in the value for the `affectedLoggers` option.
3. The default `log4j2.xml` file contains the following logger: `<logger name="com.genesyslab.platform" level="info" additivity="false">`

4. Include `com.genesyslab.platform` in the value for the `affectedLoggers` option.
5. Set the verbose option to debug.

In the sample above, the value of `affectedLoggers` should be `com.genesyslab.platform`. Error (but no debug or info) messages from Cassandra will be available in logs, and debug messages from `com.genesyslab.platform` will be available in logs.

all

Default Value: stdout
Valid Values: See table
Changes Take Effect: After start/restart

Specifies the outputs to which an application sends all log events. The log output types must be separated by a comma when more than one output is configured. For example: `all = stdout, logfile`

Valid Values (log output types):

`stdout`	Log events are sent to the Standard output (`stdout`).
`stderr`	Log events are sent to the Standard error output (`stderr`).
`network`	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.Setting the `all` log level option to the `network` output enables an application to send log events of the `Standard`, `Interaction`, and `Trace` levels to Message Server. Debug-level log events are neither sent to Message Server nor stored in the Log Database.
`memory`	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
`[filename]`	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

expire

Default Value: 3
Valid Values: See table
Changes Take Effect: After restart

Determines whether log files expire. If they do, sets the measurement for determining when they expire, along with the maximum number of files (segments) or days before the files are removed.

This option is ignored if log output is not configured to be sent to a log file.

Valid Values (log output types):

`false`	No expiration; all generated segments are stored.
`<number> file or <number>`	Sets the maximum number of log files to store. Specify a number from `1`—`1000`.
`<number> day`	Sets the maximum number of days before log files are deleted. Specify a number from `1`—`100`.

> **Warning**
> If an option's value is set incorrectly — out of the range of valid values — it will automatically be reset to `10`.

segment

Default Value: 1000
Valid Values: See table
Changes Take Effect: After restart

Specifies whether there is a segmentation limit for a log file. If there is, sets the mode of measurement, along with the maximum size. If the current log segment exceeds the size set by this option, the file is closed and a new one is created. This option is ignored if log output is not configured to be sent to a log file.

Valid Values (log output types):

`false`	No segmentation is allowed.
`<number> KB or <number>`	Sets the maximum segment size, in kilobytes. The minimum segment size is `100 KB`.
`<number> MB`	Sets the maximum segment size, in megabytes.
`<number> hr`	Sets the number of hours for the segment to stay open. The minimum number is 1 hour.

standard

Default Value: stdout
Valid Values: See table
Changes Take Effect: Immediately

Specifies the outputs to which an application sends the log events of the `Standard` level. The log output types must be separated by a comma when more than one output is configured. For example: `standard = stderr, network`

Valid Values (log output types):

`stdout`	Log events are sent to the Standard output (`stdout`).
`stderr`	Log events are sent to the Standard error output (`stderr`).
`network`	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.
`memory`	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
`[filename]`	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

time_convert

Default Value: local
Valid Values: See table
Changes Take Effect: Immediately

Specifies the system in which an application calculates the log record time when generating a log file. The time is converted from the time in seconds since 00:00:00 UTC, January 1, 1970.

Valid Values (log output types):

`local`	The time of log record generation is expressed as a local time, based on the time zone and any seasonal adjustments. Time zone information of the application’s host computer is used.
`utc`	The time of log record generation is expressed as Coordinated Universal Time (UTC).

time_format

Default Value: time
Valid Values: See table
Changes Take Effect: Immediately

Specifies how to represent, in a log file, the time when an application generates log records. A log record's time field in the ISO 8601 format looks like this: `2001-07-24T04:58:10.123`

Valid Values (log output types):

<tt>time</tt>	The time string is formatted according to the <tt>HH:MM:SS.sss</tt> (hours, minutes, seconds, and milliseconds) format.
<tt>locale</tt>	The time string is formatted according to the system's locale.
<tt>ISO8601</tt>	The date in the time string is formatted according to the ISO 8601 format. Fractional seconds are given in milliseconds.

trace

Default Value: stdout
Valid Values: See table
Changes Take Effect: Immediately

Specifies the outputs to which an application sends the log events of the `Trace` level and higher (that is, log events of the `Standard`, `Interaction`, and `Trace` levels). The log outputs must be separated by a comma when more than one output is configured. For example: `trace = stderr, network`

Valid Values (log output types):

`stdout`	Log events are sent to the Standard output (`stdout`).
`stderr`	Log events are sent to the Standard error output (`stderr`).
`network`	Log events are sent to Message Server, which can reside anywhere on the network. Message Server stores the log events in the Log Database.
`memory`	Log events are sent to the memory output on the local disk. This is the safest output in terms of the application performance.
`[filename]`	Log events are stored in a file with the specified name. If a path is not specified, the file is created in the application's working directory.

verbose

Default Value: standard
Valid Values: See table
Changes Take Effect: Immediately

Determines whether a log output is created. If it is, specifies the minimum level of log events generated. The log events levels, starting with the highest priority level, are Standard, Interaction, Trace, and Debug.

Valid Values (log output types):

`all`	All log events (that is, log events of the Standard, Trace, Interaction, and Debug levels) are generated.
`debug`	The same as `all`.
`trace`	Log events of the Trace level and higher (that is, log events of the Standard, Interaction, and Trace levels) are generated, but log events of the Debug level are not generated.
`interaction`	Log events of the Interaction level and higher (that is, log events of the Standard and Interaction levels) are generated, but log events of the Trace and Debug levels are not generated.
`standard`	Log events of the Standard level are generated, but log events of the Interaction, Trace, and Debug levels are not generated.
`none`	No output is produced.