Log Filter

Name

log-filter - filter log data.

Synopsis

`log-filter [options] [FILE]`

Description

This is a utility which allows to filter log lines and collect the statistics.

The filter, by default, skips the lines being unsupported by the format (see log format). It is possible to turn on a strict mode (`--strict`) to terminate the processing when the first wrong line appears. In this case, the utility returns no data and status code `1`.

The filter can work in two different forms *filter-and-write* and filter-and-collect*. The _filter-and-write* is used by default and a result is the same as input log but without the lines which were filtered out. The *filter-and-collect* is enabled by `-c` option and its result is described in statistics output part.

The utility processes the data given from *FILE* or, by default, its *standard input*, to the *standard output* if `-o` option is omitted.

Options

```
    -h, --help                Print this help on the standard output.

    -f, --follow              Follow log data. This option causes filter
                              to not stop when end of file is reached, but wait
                              for additional data to be appended to the input.
                              It is done if the data input is closed (i.e. file
                              became unreadable or stadard input reaches the
                              end). It also can be interrupted after the SIGINT
                              signal is received.

    -o, --out <FILE>          Determine a target output destination. This option
                              overrides the standard output which is used by
                              default.
    -s, --start <DATETIME>  Filter out the log lines that are reported before
                            the given time point. The UTC <DATETIME> has
                            "YY-MM-ddTHH:mm:ss.SSSZ" format.

    -e, --end <DATETIME>    Filter out the log lines that are reported after
                            the given time point. The UTC <DATETIME> has
                            "YY-MM-ddTHH:mm:ss.SSSZ" format.

    -p, --period <DURATION> Calculate time boundary using a relative time
                            which is used in combination with --start or
                            --end options. The <DURATION> follows an ISO-8601
                            time duration format. It is wrong to use this
                            option with --start and --end simultaneously
                            as well as without anyone of both. If a period
                            is used with a start datetime then an end datetime
                            equals the start plus the period. On the other
                            side, using a period with an end datetime means
                            start dateime equals the end minus the period.

    -l, --level <LEVEL>...  Filter the log records by their levels. It is
                            possible to pass multiple levels using a comma-
                            separated list of levels.

    -t, --thread <MASK>     Filter out the log lines which are not matched
                            with the <MASK>, where the <MASK> is a string
                            optionally containing wildcards (* matches zero
                            or more characters, ? matches exactly one
                            character).

    -m, --message <MASK>    Filter out the log lines which are not matched
                            against the <MASK>, where the <MASK> is a string
                            optionally containing wildcards (* matches zero
                            or more characters, ? matches exactly one
                            character).

    --strict                Terminate processing after the first improper
                            log line. By default, a wrong line is skipped
                            silently.

    -c, --stats             Print out collected statistics of filtered data
                            instead of data themselves. This option enables
                            filter-and-collect mode. If this option is used
                            with -f option then statistics will be printed
                            after the following accomplishes.
```

Log format

The supported log format consists of the following parts:

1. Fixed size datetime in *UTC* which follows *ISO-8601* `YY-MM-ddTHH:mm:ss.SSSZ` pattern.
2. The level literal such as `TRACE, DEBUG, INFO, WARN, ERROR`. The level is *left aligned* and always has 5 characters.
3. The thread name which is 16 characters *right aligned* string. There are two extra square brackets that enclose the thread name.
4. Size unlimited log message. The message may be empty.

All the parts are mandatory. There are single whitespaces between each part. Also, the log lines are monotonically increased sequence in terms of time.

```
19-05-23T20:57:20.564Z INFO  [            main] Starting up an application
19-05-23T20:57:25.872Z INFO  [            main] Initializing a worker pool
19-05-23T20:57:30.032Z INFO  [        worker-1] Run task #1241
19-05-23T20:57:32.421Z INFO  [        worker-2] Run task #717
19-05-23T20:57:33.843Z INFO  [            main] Connecting to storage:3121
19-05-23T20:57:58.931Z INFO  [        worker-2] Task #717 has been completed
19-05-23T20:57:59.001Z INFO  [            main] Connection established
19-05-23T20:58:14.315Z WARN  [        worker-3] Run task #467
19-05-23T20:58:15.917Z WARN  [        worker-3] Task #467 is not configured
19-05-23T20:58:34.143Z INFO  [        worker-1] Task #1241 has been completed
19-05-23T20:58:40.719Z INFO  [            main] Application started successfully
19-05-23T20:58:44.242Z INFO  [        worker-2] Run task #937
19-05-23T20:58:56.964Z ERROR [        worker-3] Task #467 finished with fail
19-05-23T20:58:57.234Z WARN  [        worker-3] Task #467 cannot be rescheduled
19-05-23T20:59:21.622Z INFO  [            main] DB health-check succeded
19-05-23T20:59:32.421Z ERROR [        worker-1] Task #8272 crashed during startup
19-05-23T20:59:33.642Z WARN  [        worker-2] Task #937 has been completed
19-05-23T20:59:33.186Z DEBUG [        worker-1] Task #8272 will be restarted
19-05-23T20:59:39.037Z INFO  [        worker-3] Run task #819
19-05-23T20:59:43.146Z INFO  [        worker-2] Run task #2911
19-05-23T21:00:19.620Z INFO  [        worker-3] Task #819 has been completed
19-05-23T21:00:20.821Z ERROR [            main] DB health-check failed
19-05-23T21:00:23.422Z INFO  [        worker-2] Task #2911 has been completed
```

Statistics output

The statistics output always has 7 rows which are

1. Number of affected log lines.
2. Average time between the log lines.
3. Numbers of lines grouped by log levels.

4. Distribution per each log levels.
5. Number of unique thread names.
6. Most encountered thread name.
7. Least encountered thread name.

```
Lines           : 200
Mean time       : PT3M34S
Levels          : TRACE 20, DEBUG 40, INFO 80, WARN 50, ERROR 10
Levels %        : TRACE 10, DEBUG 20, INFO 40, WARN 25, ERROR 5
Threads         : 12
Most thread     : main-thread
Least thread    : worker-5
```

Examples

1. Filter log lines which were reported during May 23rd, 2019:

   `log-filter -s "19-05-23T00:00:00.000Z" -e "19-05-23T23:59:59.999Z" example.log`

2. Filter log lines which have `DEBUG` or `WARN` levels and were reported under threads:

   `log-filter -l "DEBUG,WARN" -t "worker-*" example.log`

3. Filter log lines which were reported during a week since the start of May 2019:

   `log-filter -s "19-05-01T00:00:00.000Z" -p "P1W" example.log`

4. Filter log lines which were reported from 8th to 10th May 2019:

   `log-filter -e "19-05-010T00:00:00.000Z" -p "P2D" example.log`

5. Filter log lines with a message containing `failed` as a substring:

   `log-filter -m "*failed*" example.log`

6. Redirect a result to the `filtered.log` file instead of the standard output:

   `log-filter -o filtered.log example.log`

7. Filter log lines starting with `Task` in a message part and ensure all the lines are processable:

   `log-filter -m "Task*" --strict example.log`

8. Filter warning log lines and wait for new lines:

   `log-filter -l "WARN" -f example.log`

9. Read log data from the standard output filled by an abstract `log-supplier` command which periodically produces the log lines:

   `log-supplier | log-filter -t "main"`

10. Print out log statistics contained in `example.log` file:

    `log-filter -c < example.log`