Chrono-Log

Table of Contents

[Problems faced without it: 4](#_Toc348639552)

[Package details : 4](#_Toc348639553)

[Interfaces: 5](#_Toc348639554)

[Using the tool: 5](#_Toc348639555)

[Launching the tool 5](#_Toc348639556)

[Running the tool: 6](#_Toc348639557)

[Appendix A 7](#_Toc348639558)

[Configuration XML 7](#_Toc348639559)

Details about the Chrono-Log tool

# Problems faced without it:

While debugging logs and incident reports on distributed setups it is sometimes easier to debug if all the logs or selected ones can be put into one single file and lines are arranged with respect to time. This not only helps in understanding overall issue but sometimes can give insight into unexpected root causes.

Here are some use cases where you can use this tool.

1. Find if an issue with one component is affecting whole system.
2. Cases where you want to find sequence of events (messaging based operation)
3. Performance

Chrono –Log currently parses logs from:

1. Log4j
2. IIS- Webserver
3. Weblogic ACCESS logs

There are excellent aftermarket tools are available in the market .Some are open source as well.

Current tool intends to work for eGain application by means of default set of patterns included which are from egain log files.

## Package details :

The tool consists of three packages:

com.caterpillar ui.

com.caterpillar.comparators

com.caterpillar.dataobject

com.caterpillar.launcher

com.caterpillar.parsers

com.caterpillar.ui

### Interfaces:

Package: com.caterpillar.parsers

**IParsers.java**

|  |
| --- |
| **public** List<LineData> parseLog(FileData fileTuple, List<LineData> dataRepo)  **throws** FileNotFoundException; |

Its an interface that can be used to write custom parsers which are provided Out of the box. Currently there is dependency on another class in package “com.caterpillar.launcher” named “FileTypeChecker.java”. We also need to make changes here. This dependency will be removed over the period of time.

Tool loads the parsers dynamically based on configuration and hence it is independent of how many types of file that need to be processed.

## Using the tool:

Launching the tool:

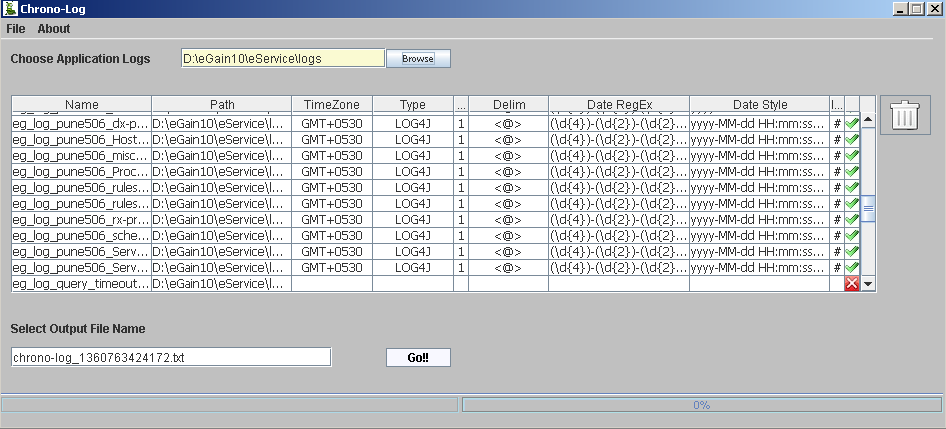
This can be done in two ways.

We can launch the tool using a batch file with following command.

|  |
| --- |
| Java –Xmx1024m –Xms1024m –jar Caterpillar.jar |

Above command also contains memory arguments. This command should be used when we want to process large number of files.

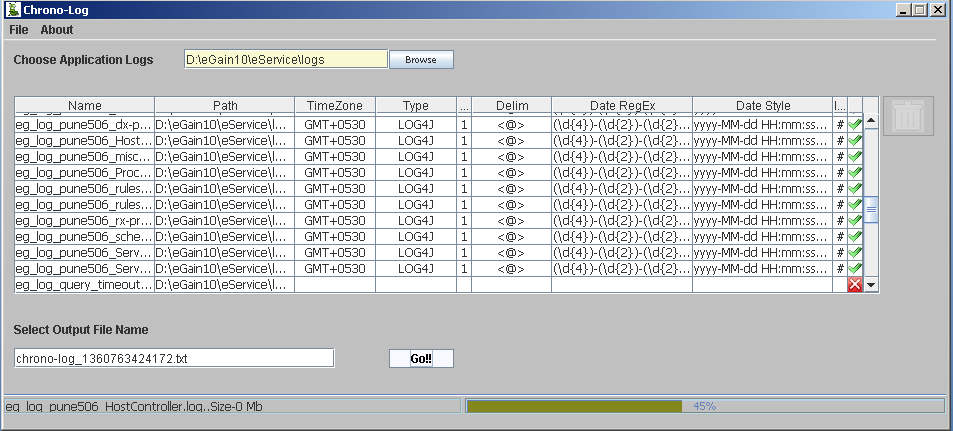
Other option is to launch the tool using “open with Java Runtime SDK” from windows context menu on windows machines. Here is the launch window with some files selected.



Important to note is that tool automatically identifies what is the file type based on some file parsing. It also marks if any file can be used or not. The last column on the grid if has a “X” it means file will not be included in the output.

### Running the tool:

Clicking on “Go” starts the processing.



Once all the files are processed, sorted data is dumped in the output file shown in screen.

The output file is created in the folder where toll is run from.

## Appendix A

### Configuration XML

|  |
| --- |
| There is configuration XML included in the jar file here is the excerpt:  Lt and gt are used for “<” and “>” respectively, we have not used standard escaping here.  <definitions>  <logtype name=*"common"* >  <acceptedfile>.log,.txt</acceptedfile>  <maxcheckcounter>10</maxcheckcounter>  </logtype>  <logtype name=*"LOG4J"* >  <regex>(\d{4})-(\d{2})-(\d{2}) (\d{2}):(\d{2}):(\d{2}).(\d{3}) (\w{3})(\+|\-)(\d{4})</regex>  <hastack>1</hastack>  <delim>lt@gt</delim>  <dateformat>yyyy-MM-dd HH:mm:ss.SSS 'GMT'Z</dateformat>  <ignoreline>#</ignoreline>  <defaultTZ>GMT</defaultTZ>  <textpattern>  <pattern>lt@gt DEBUG lt@gt</pattern>  <pattern>lt@gt ERROR lt@gt</pattern>  <pattern>lt@gt INFO lt@gt</pattern>  <pattern>lt@gt DBQUERY lt@gt</pattern>  <pattern>lt@gt PERF lt@gt</pattern>  <pattern>lt@gt TRACE lt@gt</pattern>  <pattern>lt@gt WARN lt@gt</pattern>  </textpattern>  <parserclass>com.caterpillar.parsers.Log4JParser</parserclass>  </logtype>  <logtype name=*"W3SVC"* >  <regex>(\d{4})-(\d{2})-(\d{2}) (\d{2}):(\d{2}):(\d{2})</regex>  <hastack>0</hastack>  <delim> </delim>  <dateformat>yyyy-MM-dd HH:mm:ss</dateformat>  <ignoreline>#</ignoreline>  <defaultTZ>GMT</defaultTZ>  <textpattern>  <pattern> W3SVC1 </pattern>  </textpattern>  <parserclass>com.caterpillar.parsers.W3SVCParser</parserclass>  </logtype>  <logtype name=*"WLACCESS"* >  <regex>(\d{2})/(\w{3})/(\d{4}):(\d{2}):(\d{2}):(\d{2}) (\+)(\d{4})</regex>  <hastack>0</hastack>  <delim> </delim>  <dateformat>dd/MMM/yyyy:HH:mm:ss Z</dateformat>  <ignoreline>#</ignoreline>  <defaultTZ>GMT</defaultTZ>  <textpattern>  <pattern>ime Taken:</pattern>  </textpattern>  <parserclass>com.caterpillar.parsers.AccessLogParser</parserclass>  </logtype>  </definitions> |