10/08/2020

**Log Parser**

**Version 1.1.0.27**

**User Manual**

1. **Profile**

The predefined profile DefaultProfile.xml will be used by default. There is possible to import any valid profile from the Tools->Import profile menu. The DefaultProfile.xml locates in the application work folder. Imported profile can be located in any accessible folder.

* NOTE: The Application will remember the last used profile, so the next time it will be loaded automatically.

Profile filter node example:

<Filter key="CUsbipRequest::CUsbipRequest" IsVisible ="1">

<ObjectClass>Request</ObjectClass>

<State>Created</State>

<Patterns>

<Pattern>

[%\*1d]%\*4c.%\*4c::%s [%\*7s]CUsbipRequest::CUsbipRequest: AAA %s

</Pattern>

<Pattern>

[%\*1d]%\*4c.%\*4c::%s [%\*7s]CUsbipRequest::CUsbipRequest: BBBBB %s

</Pattern>

<Pattern>

[%\*1d]%\*4c.%\*4c::%s [%\*7s]CUsbipRequest::CUsbipRequest: C %s

</Pattern>

</Patterns>

<Properties>

<Property i="0">

<PatternIndex>3</PatternIndex>

<Name>this</Name>

<DataType>string</DataType>

<Action>New</Action>

<DisplayMember>1</DisplayMember>

<FilterMember>1</ FilterMember>

</Property>

<Property i="1">

<PatternIndex>0</PatternIndex>

<Name>Time</Name>

<DataType Format ="MM/dd/yyyy-HH:mm:ss.FFF">Time</DataType>

<Action>AssignToSelf</Action>

<DisplayMember>1</DisplayMember>

<FilterMember>0</FilterMember>

<ColorKeysMember>1</ColorKeysMember>

</Property>

</Properties>

</Filter>

* 1. Filter Node

key – Internal object class identifier. Can be used as a log line delimiter for the line parsing

IsVisible – an indicator whether or not the object visible in the data grid (0 or 1)

ObjectClass – human readable object class name

State – state of the object.

* 1. Patterns Node – includes one or more parsing patterns (rules for specific log line parsing)

If the object definition has more than one syntax variations, the patterns will be processed in the sequence while one of the patterns will give valid result.

* 1. Properties Node – list of the object dynamic properties.
  2. Property Node

i="n" – the sequence number of property. It defines the order of log line parsing processing

PatternIndex – index of property value in the pattern

Name – property name

DataType – property data type

Format ="xxx"> – property data format (used for conversion of date/time/numeric types represented as string in a log line to the appropriated original data types)

Action – required property action. The possible values of action

* + New – creates a new parsing object
  + AssignToSelf – creates a property and/or assigns the parsed value to the property
  + AssignDataBuffer – assigns a data buffer records from the log to the property
  + Locate – locates an already created parsing object by value
  + Assign – assigns a value to located object (NOTE: The “Assign” action always following after the “Locate” action and refers to it.

DisplayMember – an indicator whether or not the property value will be displayed as part of description on the state object surface (0 or 1).

FilterMember – an indicator whether or not the property value can be filtered by Custom Filter (0 or 1).

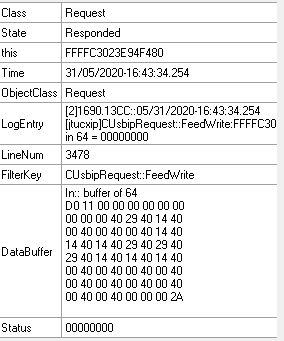
ColorKeysMember – an indicator whether or not the property value used in object state colorizing algorithm (0 or 1).

.

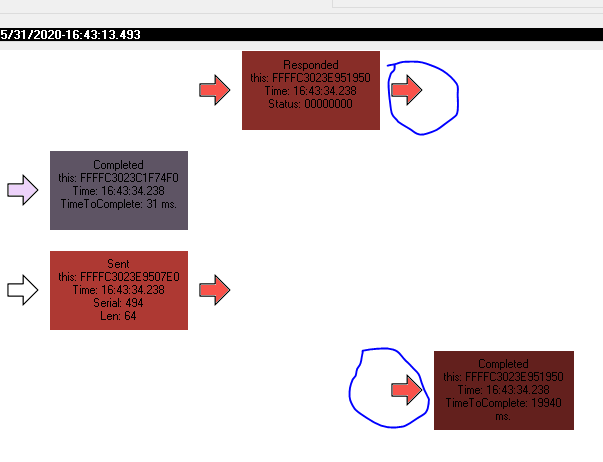
1. **Main Application Form**
   1. Menu File ->Load LOG File opens a LOG file for loading (or shortcut CTRL-L).
   2. The “Device” combo box allowing to select the required device for display



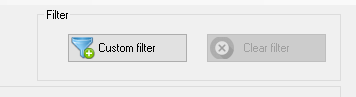
* 1. Info box displays all properties information of selected state object:



* 1. Colorized arrows between the status objects are clickable and mouse click will move selection to the next/previous state of “interrupted” chains. (NOTE: White arrows are not clickable!)



* 1. “Custom Filter” button will open the filter dialog form. “Clear filter” will clear a current filter



* 1. Tools menu.

Tools -> Import profile allows to open an alternative profile.

Tools -> Edit current profile will open a current application profile

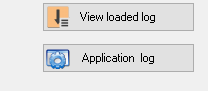
Tools -> Pattern validation tool allows to test a pattern expression in a profile

* 1. “View loaded log” button will open the current log file in an external editor. By default this is Windows *Notepad* text editor.

If the *Notepad++* defined as external editor data grid has any selected object, the external editor will be

opened on the log line appropriated to the this object.

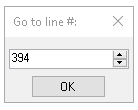
“Application log” button will open the application log for current session.



1. **Go to line number form**

Allow user to search a specific log line representation in the loaded data grid. Calling from menu:

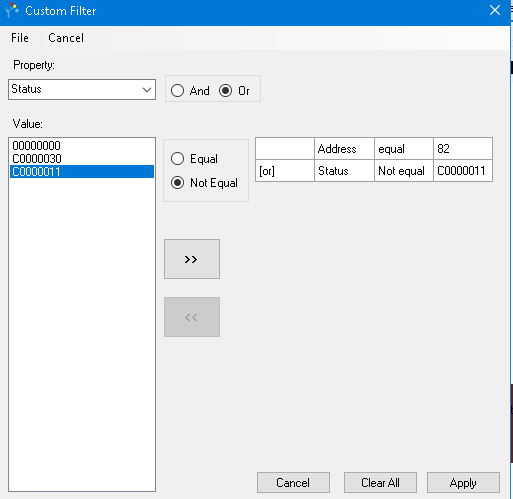
Edit->Go to line… (or from the shortcut CTRL-G)



1. **Custom Filter form**

Allow filtering of the currently loaded data grid by one or more logical criteria. The filtering bases on the existing properties values. The values represented per current device, selected on the main form. If no specific device selection (All devices… selected) the values will be represented for all available devices.

* NOTE: Logical grouping or parentheses currently is not supported! So the priority of logical operations will be proceed according to the standard Boolean operations order.



* Once defined filter can be saved and loaded later as file via File -> Save filter / Load filter menu
* After the custom filter was applied, there is possible to view the current filter content on the tooltip of the “Custom filter” button (Main form)

1. **Caching data**

Once successfully loaded view data from any log file will be saved in a binary cache pool. If cache exists, it will be loaded automatically, instead the original log file. That is for better performance purposes. All caches will be placed in the same directory according to the current configuration. There is possible to delete any existing cache any time from the main application menu (File->Clear cache)

1. **Configuration**

The application configuration file LogParserApp.exe.config located in it working folder.

The typical configuration example:

<appSettings>

<add key="AppLog" value="Application.log"/>

<add key="AppLogIsActive" value="1"/>

<add key="ExternalEditorExecutablePath" value="notepad++.exe"/>

<add key="ExternalEditorArguments" value="-ro -nosession -notabbar"/>

<add key="MaxLoadLines" value="50000"/>

<add key="Colors" value = "#87cefa, #F9524A, #37FB02, #FB00FF,#EDD2FA, #D2FAFA"/>

<add key="DisplayInInfobox" value = "this, Parent, State, Line, LineNum, Port, ID"/>

<add key="ColorCorrectionFactorPercent" value = "15"/>

<add key="MaxVisualDescriptionLength" value="30"/>

<add key="VisualTimeFormat" value="HH:mm:ss.FFF"/>

<add key="VisualDateTimeFormat" value="{0:dd/MM/yyyy-HH:mm:ss.FFF}"/>

<add key="CachePoolPath" value="\Cache\"/>

</appSettings>

* AppLog – absolute or relative path for application log.
* AppLogIsActive – indicates whether or not application log active (0 or 1).
* ExternalEditorExecutablePath – absolute or relative path for external editor executed.
* ExternalEditorArguments – external editor optional parameters (if applicable)
* MaxLoadLines – maximum number of log lines to loading.
* Colors – Comma delimited string of hexadecimal color codes for state objects presentation .
* DisplayInInfobox – Comma delimited string of properties for InfoBox presentation.
* ColorCorrectionFactorPercent – Percent of one step color correction in the object state chain (from light to darker).
* MaxVisualDescriptionLength – Maximum length of the single line in the state object description. Used for the true word wrapping of the description.
* VisualTimeFormat – Standard time format for the time stamp presentation.
* VisualDateTimeFormat – Standard date and time format for the time stamp presentation.
* CachePoolPath – absolute or relative path for actual application cache pool.

**Enjoy and sorry for my terrible English!**