Soft Properties: Supported Parsing Operators, Rules, and Regular Expressions

These topics provide listings of the operators, rules, and expressions that are supported for creating soft properties.

- Supported Parsing Rules and Operators, page A-1
- Supported Regular Expressions, page A-3

For information on supported triggers for TCAs, see Creating a New Threshold-Crossing Alarm, page 7-1.

Supported Parsing Rules and Operators

Table A-1 lists the predefined text manipulation operators available for parsing raw device input and turning each into a soft property that is available in the Add/Edit Parsing Rule dialog box.

Table A-1 Supported Parsing Operators and Rules for Soft Properties

Operator	Description	Parameter	Parameter Description	Parameter Validation Rules
Header and Footer	Removes a specified number of lines from the header and footer of the input text.	Header lines	Number of header lines to be removed.	Integers only. Mandatory.
		Footer lines	The number of footer lines to be removed.	Integers only. Mandatory.
Remove Lines	Removes a range of lines from the specified starting row to the specified end row the input text.	From line	Index of first row to begin removal, inclusive.	Integer only. Mandatory.
		To line	Index of last row to be removed, inclusive.	Integer only; must be equal to or greater than *From line*. Mandatory.
Select Lines	Extracts a range of lines from the specified starting row to the specified end row f the input text.	From line	Index of first row to begin selection, inclusive.	Integer only. Mandatory.
		To line	Index of last row to be selected, inclusive.	Integer only; must be equal to or greater than *From line*. Mandatory.

Table A-1 Supported Parsing Operators and Rules for Soft Properties (continued)

Operator	Description	Parameter	Parameter Description	Parameter Validation Rules
Replace	Finds one substring or all occurrences of a substring that matches a specified regular expression, and replaces it with a specified value.	Expression	Search for value or regular expression.	Text. Mandatory.
		With	Replace string with value or regular expression.	Text. Mandatory.
		From Index	Starting index.	Integer; must be 1 or higher. Mandatory.
		Replace All	Check box. Select this option to replace all occurrences of the matching substrings, otherwise only the first instance is replaced.	Default is unchecked.
Match	Finds and extracts a substring that matches a specified regular expression. If no match can be found, the output buffer receives an empty string or receives the original input, based on choice in *If no match* drop down list.	Expression	Search for value or regular expression.	Text. Mandatory.
		If no match	Returns the original input or and empty string, based on your choice.	
Set	Prints the results of the input and output buffers.	Expression	Regular expression template to use for formatting. $_$ specifies the main output buffer.	Text. Mandatory.
Substring	Extracts a substring of a specified length from a specified starting point.	From Index	Begin index to select.	Integer; must be 1 or higher. Mandatory.
		Length	How many characters to select.	Integer. Mandatory.
Parse Integer	Uses the substring rule. When a result is received with the substring, it is converted into an integer value. **Note** If the substring operator contains any characters, the parsing integer operator fails.	From Index	Starting index to select.	Integer; must be 1 or higher. Mandatory.
		To Index	Ending index to select.	Integer. Mandatory.

Supported Regular Expressions

A regular expression consists of a character string in which some characters are given special meaning with regard to pattern matching. These topics are based on the documentation of the package GNU RegExp. These tables list the supported and unsupported expressions.

Note Some flavors of regular expression utilities support additional escape sequences. The following is not meant to be an exhaustive list. In the future, gnu.regexp might support some or all of the following but they are currently unsupported:

(**?mods) —** Inline compilation/execution modifiers (Perl5)

\G — End of previous match (Perl5)

[*.symbol*.] — Collating symbol in class expression (POSIX).

- **[=*class*=] —** Equivalence class in class expression (POSIX)
- **s/foo/bar —** Style expressions as in sed and awk

Table A-2 Supported Positional Operators

Character	Description
^	Matches at the beginning of a line.
$	Matches at the end of a line.
\A	Matches the start of the entire string.
\Z	Matches the end of the entire string.
\b	Matches at a word break (Perl5 syntax only).
\B	Matches at a nonword break (opposite of \b) (Perl5 syntax only).
\<	Matches at the start of a word (egrep syntax only).
\>	Matches at the end of a word (egrep syntax only).

Table A-3 Supported One-Character Operators

Character	Description
.	Matches any single character.
\d	Matches any decimal digit.
\D	Matches any nondigit.
\n	Matches a newline character.
\r	Matches a return character.
\s	Matches any whitespace character.
\S	Matches any nonwhitespace character.
\t	Matches a horizontal tab character.
\w	Matches any word (alphanumeric) character.

Table A-3 Supported One-Character Operators (continued)

Character	Description
\W	Matches any nonword (alphanumeric) character.
x	Matches the character *x*, if *x* is not one of the above listed escape sequences.

Table A-4 Supported Character Class Operators

Character	Description
[*abc*]	Matches any character in the set *a*, *b*, or *c*.
[^*abc*]	Matches any character not in the set *a*, *b*, or *c*.
[*a-z*]	Matches any character in the range *a* to *z*, inclusive.
Leading or trailing dash	Interpreted literally.

Table A-5 Supported Character Class Operators—if RE_CHAR_CLASSES Is Enabled

Character Sequences	Description
[:alnum:]	Any alphanumeric character.
[:alpha:]	Any alphabetic character.
[:blank:]	A space or horizontal tab.
[:cntrl:]	A control character.
[:digit:]	A decimal digit.
[:graph:]	A nonspace, noncontrol character.
[:lower:]	A lowercase letter.
[:print:]	Same as graph, but also space and tab.
[:punct:]	A punctuation character.
[:space:]	Any whitespace character, including newline and return.
[:upper:]	An uppercase letter.
[:xdigit:]	A valid hexadecimal digit.

Table A-6 Supported Subexpressions and Back References

Expression	Description
(*abc*)	Matches whatever the expression *abc* would match, and saves it as a subexpression. Also used for grouping.
(?:...)	Pure grouping operator; does not save contents.
(?#...)	Embedded comment; ignored by engine.
n	Where 0 < *n* < 10, matches the same thing the *n*th subexpression matched.

Table A-7 Supported Branching (Alternation) Operator

Expression	Description
a|*b*	Matches whatever the expression *a* or *b* would match.

Table A-8 Supported Repeating Operators (Operate on Previous Atomic Expression)

Symbols	Description
Note	If any of these operators are immediately followed by a question mark (?), the repeating operator stops at the smallest number of repetitions that can complete the rest of the match (Stingy (Minimal) Matching).
?	Matches the preceding expression or the null string.
*	Matches the null string or any number of repetitions of the preceding expression.
+	Matches one or more repetitions of the preceding expression.
{*m*}	Matches exactly *m* repetitions of the one-character expression.
{*m*,*n*}	Matches between *m* and *n* repetitions of the preceding expression, inclusive.
{*m*,}	Matches *m* or more repetitions of the preceding expression.

Table A-9 Supported Lookahead

Expression	Description
Note	Lookahead refers to the ability to match part of an expression without consuming any of the input text. The variations in this table are supported.
(?=*foo*)	Matches at any position where *foo* would match, but does not consume any characters of the input.
(?!*foo*)	Matches at any position where *foo* would not match, but does not consume any characters of the input.