Regex component 1.0

This component includes a group of commands that let you do operations on text by using regular expressions (in replacement of the Regex commands of QFree©).

The commands provided are:

Rgx_MatchText	Matches a regular expression against a text variable
Rgx_SubstituteText	Substitutes all matches of a regular expression against a text variable
Rgx_SplitText	Splits a text variable into segments separated by matches of a regular expression
Rgx_ExtractText	Extracts all matches of a regular expression against a text variable

Notes :

- Some changes (in red) was made for the parameters type (array pointer as array name) because of the components' architecture.
- The QFree's constants was not in the components and must be replaced with numerical value.
- It was only possible for this version to manage 4 regex options of QFree©. See Regex options
- The none text-based regex commands was not rewriten because of the new capabilities of 4D v11.

Rgx_MatchText

Rgx_MatchText(patternText; targetText{; matchedSegments{; regexFlags}}) -> result			
->	patternText	Text	Regular expression
->	targetText	Text	Target text
<-	matchedSegments	Pointer	Array of extracted segments
->	regexFlags	Longint	Regular expression flags
<-	error	Longint	Error result

Matches a regular expression against a text variable.
Parameter **patternText** is the regular expression to be matched.
Parameter **targetText** is the target text variable.
If a non nil pointer is passed as third parameter, match results are returned in the **matchedSegments** text array. Element 0 of matchedSegments will contain the whole pattern match, and the other elements will contain all matched subpatterns.
Optional parameter **regexFlags** specifies a set of flags that control the matching operation.
Use the regex options by combining them with bitwise "or".

Rgx_SubstituteText

Rgx_SubstituteText(patternText; replacementText; targetText{; regexFlags}}) -> result			
->	patternText	Text	Regular expression
->	replacementText	Text	Replacement text
<->	targetText	Pointer	Target text
->	regexFlags	Longint	Regular expression flags
<-	error	Longint	Error result

Substitutes all matches of a regular expression in a text variable with a replacement string.
Parameter **patternText** is the regular expression to be matched.
The replacement string is specified by the **replacementText** parameter. The replacement

string may contain group references in the \digit form. Each backref is substituted by the corresponding sub-pattern match (\0 is the whole pattern match, \1 the first group, \2 the second etc.).
Other special character sequences that can be used in the replacement string are: See http://www.icu-project.org/userguide/regexp.html
Parameter **targetText** is the target text variable.
Optional parameter **regexFlags** specifies a set of flags that control the matching operation.
Use the regex options by combining them with bitwise "or".

Rgx_SplitText

Rgx_SplitText(patternText; targetText; splitSegments{; regexFlags}) -> result			
->	patternText	Text	Regular expression
->	targetText	Text	Target text
<-	splitSegments	Pointer	Array of text segments
->	regexFlags	Longint	Regular expression flags
<-	error	Longint	Error result

Splits the target into segments separated by matches of the specified regular expression. If grouping parentheses are used in the pattern, then the text of all matched groups in the pattern are also added to the segment list.
Parameter **patternText** is the regular expression to be matched.
Parameter **targetText** is the target text variable.
Split segments are returned in the **splitSegments** array.
Optional parameter **regexFlags** specifies a set of flags that control the matching operation.
Use the regex options by combining them with bitwise "or".

Rgx_ExtractText

Rgx_ExtractText(patternText; targetText; groupsToExtract; extractedMatches{; regexFlags}) -> result			
->	patternText	Text	Regular expression
->	targetText	Text	Target text
->	groupsToExtract	Text	Group numbers to extract
<-	extractedMatches	Pointer	Array of extracted segments
->	regexFlags	Longint	Regular expression flags
<-	error	Longint	Error result

Extracts all matches of a regular expression against a text variable.
Parameter **patternText** is the regular expression to be matched.
Parameter **targetText** is the target text variable.
Parameter **groupsToExtract** specifies the groups to be extracted. If it is empty or "0", only the whole pattern matches are extracted. If the pattern contains grouping parentheses, the groupsToExtract can be a list of group numbers to be extracted. For example, by specifing "2 1", all matches of the second and first sub-pattern will be extracted in the specified order.

Parameter **extractedMatches** receives the pattern matches. Depending on the groups specified in groupsToExtract, this parameters can be:
• 1-dimension text array, if groupsToExtract specifies a single group
• 2-dimension text array, if groupsToExtract specifies a list of groups

Optional parameter **regexFlags** specifies a set of flags that control the matching operation.
Use the regex options by combining them with bitwise "or".

Regex options

0x0001	Letters in the pattern match both upper and lower case letters
0x0002	The "start of line" and "end of line" constructs match immediately following or immediately before any newline in

0x0002	the subject string, respectively, as well as at the very start and end.
0x0004	A dot metacharater in the pattern matches all characters, including newlines. Without this option, newlines are excluded.
0x0008	Whitespace data characters in the pattern are totally ignored except when escaped or inside a character class, and characters between an unescaped # outside a character class and the next newline character, inclusive, are also ignored. This makes it possible to include comments inside complicated patterns.
0x0400	For Rgx_SplitText : Skeep empty lines
0x0800	For Rgx_SplitText : Trim unnecessary whitespace or tab from the start and the end of a string.