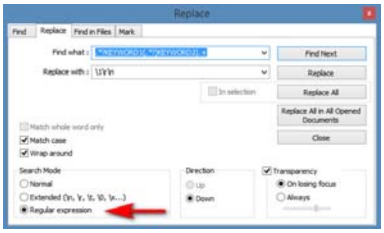
MY LITTLE REGEX COOKBOOK

For users of the free Notepad++, aka Notepad npp.

September 2020. Version 1.3

Some useful regex commands for Notepad++ — tested and working!

I assume 'Regular Expression' is ticked:



Find what : ^.*KEYWORD1(.*?)KEYWORD2.*

Replace with : \1\r\n

[ ] In selection

[ ] Match whole word only
[✓] Match case
[✓] Wrap around

Search Mode
○ Normal
○ Extended (\n, \r, \t, \0, \x...)
● Regular expression

Direction
○ Up
● Down

[✓] Transparency
● On losing focus
○ Always

STRIP:

Strip a text file of all its HTML tags:

Find: </?[a-z][a-z0-9]*[^<>]*>

This shorter and more memorable version will also work, and will also clear unwanted items in the HTML page source such as as <!-- -->

Find: <[^>]+>

Strip a file of nearly all HTML tags (but leave the Web links intact):

Find: <(?!\/?a(?=>|\s.*>))\/?.*?>

DELETE IN A LIST:

Delete lines in a list if they contain this keyword or character in them:

Find: .*DELETE.*

Delete lines if they contain any one of multiple keywords or characters:

Find: `.*CATS|DOGS|SHEEP.*`

Keep only the first word in each line, delete all else:

Find: `(?-s)\h.+`

Delete everything on a line, if after the] character:

Find: `\].*$`

(Ensure the ‘matches newline’ switch is OFF for this).

SPACING:

Delete excess blank lines (and without jamming everything together):

Find: `^([ \t]*)\r?\n\s+$`

Replace: `\1`

Remove all blank spaces between words, if larger than one space:

Find: `[ \t]{2,}`

Replace: `\1`

KEEP LINES IN A LIST:

Keep only the lines which START with these keywords or characters, and delete all other lines:

Find: `^(?!CATS|DOGS|SHEEP).+`

Keep only the lines that contain ANYWHERE one of these three different keywords, delete all other lines:

Find: `^((?!CATS|DOGS|SHEEP).)*$`

DELETE BETWEEN AND AROUND:

Delete everything except what is between these words, and also delete the search words themselves:

Find: `.*LEFT-KEYWORD(.*)RIGHT-KEYWORD.*`

Replace: `\1`

Add a \ before the KEYWORD, if the word is also a regex operator such as |

Delete everything that {{sits between double-curly brackets}} on any line:

Find: `\{\{[^\}]*\}\}`

Delete a line if it contains a keyword, and also delete the two lines AFTER it:

Find: `(?-s)^.*DELETEWORD.*\R(.*\R){2}`

Delete a line if contains a keyword, and also delete the two lines BEFORE it:

Find: `(?-s)(.*\R){2}.*DELETEWORD.*\R`

Delete everything that sits between two keywords, across multiple lines:

Find: `(?<=STARTWORD)(.*)(?=ENDWORD)`

Keep only the first string between two keywords, delete everything else:

Find: `.*STARTWORD ((?s:.*?))ENDWORD.*`

Replace: `\1`

Works even if there are further occurrences of STARTWORD blah ENDWORD in the text. Note also the space after STARTWORD in the regex.

SWOP TEXT AROUND IN A LINE:

Simple swap of words, found anywhere in any line:

Find: `(word1);(word2)`

Replace: `$2:$1`

Move any text in [[double square brackets]] to the end of the line it is in.

Find: `(?-s)(\[\[.*\]\])[\h]*(.*?)[\h]*$`

Replace: `$2 $1`

MARK MULTIPLE LINES:

Select multiple blocks between two keywords, and highlight all these blocks:

Find: `(?<=STARTINGWORD)([\s\S]*?)(?=ENDWORD)`

This one works in Find & Mark, not in Search and Replace, and 'Bookmark Line' must be ticked. It adds only one of Notepad++'s Bookmark 'blue buttons', at the top of each marked and highlighted block.

In practice this is only useful for visual recall when scrolling — since it seems nothing on earth can export the resulting marked text. I found no way to tell Notepad++ to "expand all marks by n lines", which would mean that the highlighted blocks could be copied and exported.

SELECT AND COPY MULTIPLE LINES:

This simply selects a single substring of text that sits between two keywords, rather than marking it:

This one works in simple Find, not in Search and Replace, or in Mark.

Find: (?<=STARTWORD)(.*)(?=ENDWORD)

Once this regex has selected the block of text, that text should display as being highlighted for the user. This method does allow the Notepad++ user to simply press Ctrl + C to copy the selected text to the clipboard. Both of the above methods can also work with phrases.

EXTRACT

EXTRACT A SIMPLE LIST FROM STRUCTURED DATA, ONE ITEM PER LINE:

Your source file has multiple instances of a repeating label. Each time this label has some variable value after it. For instance...

"username":"cat99";"email":"blah@blah.com";"username":"dog33";"emai l":"blah@blah.com";"username":"goldfish75";"email":"blah@blah.com";

You wish to extract a simple list of these variable usernames, one name per line, thus..

cat99

dog33

goldfish75

In Search & Replace, with 'Matches newline' ticked:

Find: (?s-i).+?"(username":"*".+?)"|.+

Replace: \1\r\n

OR Replace: Username: \1\r\n for result Username: Cat99

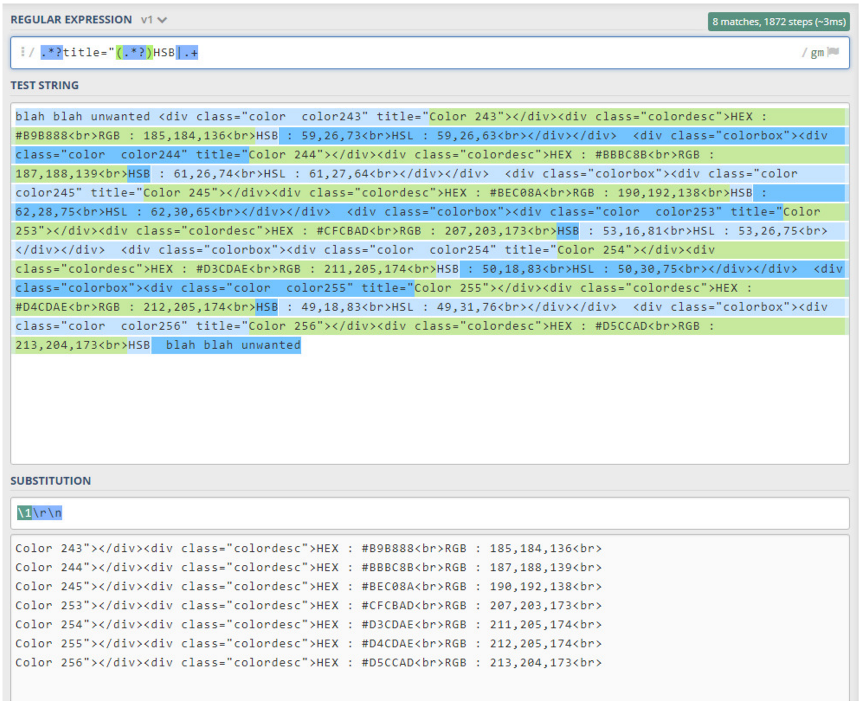
EXTRACT ALL SUBSTRINGS FOUND BETWEEN TWO STRINGS, PUT INTO LIST:

Find: .*?KEYWORD1(.*?)KEYWORD2|.+

Replace: \1\r\n

‘Matches newline’ is ticked.

The list will not include the two keywords (aka ‘delimiters’) themselves.



REGULAR EXPRESSION v1

8 matches, 1872 steps (~3ms)

```
.*?title="(.*?)HSB|.+
```

/ gm

TEST STRING

```
blah blah unwanted <div class="color  color243" title="Color 243"></div><div class="colordesc">HEX :
#B9B888<br>RGB : 185,184,136<br>HSB : 59,26,73<br>HSL : 59,26,63<br></div></div>  <div class="colorbox"><div
class="color  color244" title="Color 244"></div><div class="colordesc">HEX : #BBBC8B<br>RGB :
187,188,139<br>HSB : 61,26,74<br>HSL : 61,27,64<br></div></div>  <div class="colorbox"><div class="color
color245" title="Color 245"></div><div class="colordesc">HEX : #BEC08A<br>RGB : 190,192,138<br>HSB :
62,28,75<br>HSL : 62,30,65<br></div></div>  <div class="colorbox"><div class="color  color253" title="Color
253"></div><div class="colordesc">HEX : #CFCBAD<br>RGB : 207,203,173<br>HSB : 53,16,81<br>HSL : 53,26,75<br>
</div></div>  <div class="colorbox"><div class="color  color254" title="Color 254"></div><div
class="colordesc">HEX : #D3CDAE<br>RGB : 211,205,174<br>HSB : 50,18,83<br>HSL : 50,30,75<br></div></div>  <div
class="colorbox"><div class="color  color255" title="Color 255"></div><div class="colordesc">HEX :
#D4CDAE<br>RGB : 212,205,174<br>HSB : 49,18,83<br>HSL : 49,31,76<br></div></div>  <div class="colorbox"><div
class="color  color256" title="Color 256"></div><div class="colordesc">HEX : #D5CCAD<br>RGB :
213,204,173<br>HSB  blah blah unwanted
```

SUBSTITUTION

```
\1\r\n
```

```
Color 243"></div><div class="colordesc">HEX : #B9B888<br>RGB : 185,184,136<br>
Color 244"></div><div class="colordesc">HEX : #BBBC8B<br>RGB : 187,188,139<br>
Color 245"></div><div class="colordesc">HEX : #BEC08A<br>RGB : 190,192,138<br>
Color 253"></div><div class="colordesc">HEX : #CFCBAD<br>RGB : 207,203,173<br>
Color 254"></div><div class="colordesc">HEX : #D3CDAE<br>RGB : 211,205,174<br>
Color 255"></div><div class="colordesc">HEX : #D4CDAE<br>RGB : 212,205,174<br>
Color 256"></div><div class="colordesc">HEX : #D5CCAD<br>RGB : 213,204,173<br>
```

Demo for extraction between two strings, and output as list.

EXTRACT FROM BETWEEN TWO KEYWORDS, AND PUT RESULTS INTO A LIST (and **include** the original keywords in the output list):

Find: .*?KEYWORD1(.*?)KEYWORD2|.+

Replace: KEYWORD1\1 KEYWORD2\r\n

‘Matches newline’ is ticked.

EXTRACT ALL LINES NOT CONTAINING A KEYWORD, TO A NEW LIST:

Find: ^(?!.*KEYWORD).*$

Replace: \r\n

OUTPUT A LIST

Variant ‘Replace’ switches, used when extracted substrings in a list format.

Your regex finds substrings that need to become a simple list, one per line:

Replace: \1\r\n

Your regex finds substrings that need to output as a tab-separated list:

Replace: \1\t

Your regex finds substrings that need to be in a list with each line "wrapped in quotes" (or some other container, such as commas):

Replace: "\1"\r\n

This method is a clunky-but-easy way to re-introduce your keywords to each item on your list. For example, you use CAT and DOG as strings, and you capture everything between them. By outputting the results to a list thus...

Replace: CAT\1DOG\r\n

... the strings (search words) are re-introduced to the list output, and are placed at the correct point in each line. This method can also be used to ‘top and tail’ output by adding quite complex additions to starts and ends.

(Regex cannot be used to number lines 1,2,3,4, since it has no maths engine. Use the genuine freeware ‘List Numberer’).

"MY LIST IS TOO CRAMPED!" SPACE OUT A NEW LIST, BY INSERTING NEW BLANK LINES ON EVERY OTHER LINE:

Find: (.)$

Replace: $1\n

SIMULATE LINE-BREAKS IN MULTLINE SEARCH/REPLACE:

Replace: \nKEYWORD

Where \n represents the line-break needed in your search/replace string.

For this to work you have to make sure to select "Extended" search mode in the bottom left corner of the search window.

This is especially useful for macros, which may not record the use of older Notepad++ plugins such as ToolBucket (and its handy user-friendly multiline search-replace boxes).

ends