**RegExp Object reference**

REGEXP

regular expressions - formulas

/pattern/modifiers

stringToSearch.search(formula) - searches and returns the index

stringToSearch.match(formula) - searches and returns an array

regExpVar.exec(stringToSearch) - same as above but it returns only one result after each execution

stringToSearch.replace(formula, "forWhat"); - replacing things in formula by "forWhat"

formula.test(stringToSearch); - testing if something from formula exists in stringToSearch

Modifiers:

g - global - searching through full string

i - insensitive - case Insensitive

What formulas (patterns) can you create using RegExp?

-----------------------------------------------------------------------------

. - any character

formula: /.ag/ - it will find rag, bag, 4ag, $ag etc.

formula: /A..k/ - it will find Arek A4Gk, etc.

-----------------------------------------------------------------------------

\* - Matches any string that contains zero or more occurrences of the preceding character

formula /M\*arek/i - it will match Arek, arek, Marek, marek, mmarek,MMarek MMarek, MmMmmmmmarek ...

-----------------------------------------------------------------------------

+ - Matches any string that contains one or more occurrences of the preceding character

conclusion: it requires preceding character at LEAST once.

formula /fe+d/ matches feed and fed but doesn't match fd

-----------------------------------------------------------------------------

? -Matches any string that contains zero or one occurrences of the preceding character

conclusion: the character can exist but it doesn't need to

formula /M?arek/ - it will match Arek, arek, Marek, marek.

-----------------------------------------------------------------------------

{n} - Matches any string that containts EXACTLY n occurences of the preceding character

formula /Zo{2}/ - it will match only Zoo.

-----------------------------------------------------------------------------

{n,} - Matches any string that containts n or more occurences of the preceding character

formula /Zo{2,}/ - it will match Zoo, Zooooo, Zoooooooo

-----------------------------------------------------------------------------

{n,m} - Matches any string that containts minimally n occurences or maximally m occurences of preceding character

formula/Zo{2,4}/ - it will match Zoo, Zooo, Zoooo

-----------------------------------------------------------------------------

^ - Only matches the beginning of a string

formula /^the/i - it will match "The hole" it won't match "In the hole";

-----------------------------------------------------------------------------

$ - Only matches the end of a string

formula /g$/i - it will match only string that ends with 'g' character

-----------------------------------------------------------------------------

[ao] - Match any of the character enclosed in the character set. (think as it it was a single character to match)

for example /A[BT]C/ - matches ABC and ATC - doesn't match ABTC or AKC

-----------------------------------------------------------------------------

[^ao] -

for example /A[^BT]C/ - matches everything EXCEPT ABC and ATC so it matches ADC, AKC etc.

-----------------------------------------------------------------------------

[a-d] - matches any single character in a scope from 'a' to 'd'

so a, b, c, d

Conclusion: [0-9] - any number - same as \d

[a-z] - any small case character

[A-Z] - any large case character

[a-zA-Z] - any character

/w - any character [A-Za-z0-9\_]

-----------------------------------------------------------------------------

[^a-d] - matches every single character except a,b,c,d

-----------------------------------------------------------------------------

\ - this character allows you to interpret reserved character like \*, ., ^ etc. as a character to search for

formula /f\\*\\*\\*/ will match f\*\*\*

-----------------------------------------------------------------------------

\s - white character

-----------------------------------------------------------------------------

(x) - saves x;

np.

var d = "ViolaArkadiusz";

var e = d.replace(/(V)(i)(o)(la)/gi, "$4$3$2$1");

results in: laoiVArkadiusz

([a-z][0-9])+ any number of occurences like a4j5j5n5

[a-z][0-9]+ means: a1241241241241241

-----------------------------------------------------------------------------

x(?=y) - will match x only if y is after the x

var a = "Arek Wlodarczyk, Arkadiusz Kowalski, Arek Nowak";

var b = a.replace(/Arek.(?=Wlodarczyk)/gi, ""); - will remove Arek with Wlodarczyk surname because it's only name with Wlodarczyk after Arek

-----------------------------------------------------------------------------

x(?!y) - same as above x(?=y), but it will find x only if there is no y

-----------------------------------------------------------------------------

x|y - matches x or y

great for checking extensions: /jpg|gif/gi - will match = "landscape.GIF" the GIF word.