Regex — To Match or Not to Match

Marcin Sosnowski

2018-10-12

What is

Basic Syntax

Character	Behavior
/.../	Starting and ending regex delimiters
|	Alternation
()	Grouping

Wildcards/Character Class Shorthands

Character	Behavior
.	Match any character
\w	Match a word character (alphanumeric plus "_")
\W	Match a non-word character
\s	Match a whitespace character
\S	Match a non-whitespace character
\d	Match a digit character
\D	Match a non-digit character

Character Classes

Character	Behavior
[...]	Match a character in the brackets
[^...]	Match a character not in the brackets
[a-z]	Match a character in the range a to z

Position Matching

Character	Behavior
^	Match beginning of line
$	Match end of line
\b	Match word boundary
\B	Match non-word boundary

Repetition Factors

(greedy, match as many times as possible)

Character	Behavior
*	Match 0 or more times
+	Match 1 or more times
?	Match 1 or 0 times
{n}	Match exactly n times
{n,}	Match at least n times
{n,m}	Match at least n but not more than m times

Advanced Syntax

Character	Behavior
non-meta character	Match character
{}[]()^ $.|*+?\	Metacharacters, to match these characters, override (escape) with \
\	Override (escape) next metacharacter
\n	Match capture buffer *n*
(?:...)	Non-capturing group

Lazy Repetition Factors

(match minimum number of times possible)

Character	Behavior
*?	Match 0 or more times
+?	Match 1 or more times
??	Match 0 or 1 time
{n}?	Match exactly n times
{n,}?	Match at least n times
{n,m}?	Match at least n but not more than m times

Look-Ahead and Look-Behind

Character	Behavior
(?=...)	Zero-width positive look-ahead assertion. E.g. *regex1* (?=*regex2*), a match is found if both *regex1* and *regex2* match. *regex2* is not included in the final match.
(?!...)	Zero-width negative look-ahead assertion. E.g. *regex1* (?!*regex2*), a match is found if *regex1* matches and *regex2* does not match. *regex2* is not included in the final match.
(?<=...)	Zero-width positive look-behind assertion. E.g. (?<=*regex1*) *regex2*, a match is found if both *regex1* and *regex2* match. *regex1* is not included in the final match.
(?<!...)	Zero-width negative look-behind assertion.

DATE OPPERATIONS

Add hyphens between date parts

date='20181012';
date='2018';
date='201810';
date='2018-10';

```
if length(date) eq 8 then do;
    newDate = substr(date,1,4)||'-'||substr(date,5,2)||'-'||substr(date,7,2);
  end;
  else if length(date) eq 6 then do;
    newDate = substr(date,1,4)||'-'||substr(date,5,2);
  end;
  else do;
    newDate=date;
  end;
```

```
newDate=prxchange ('s/(\d{4})(\d{2})/$1-$2/',1,strip(date));
newDate=prxchange ('s/(\d{4}-\d{2})(\d{2})/$1-$2/',1,strip(newDate));
```

ISO DATE

```
_isodt=prxparse('/^([0-9]{4})(-)(1[0-2]|0[1-9])(-)(3[01]|0[1-9]|[12][0-9])$/');/
 ASTDT=(input(AESTDTC,B8601DA.)


_isodttm=prxparse('/^([0-9]{4})(-)(1[0-2]|0[1-9])(-)(3[01]|0[1-9]|[12][0-9])(T)(2[0-
3]|[01][0-9])(:)([0-5][0-9])$/’);
ASTDTM=input(AESTDTC,E8601DT.);
```

DATE OPPERATIONS

Fri OCT 12 04:56:33 2016

12OCT2016:04:56:33 (datetime18.)

```
Date=prxchange('s/([A-Za-z]{3})(\s)([A-Za-z]{3})(\s{1,2})(\d{1,2})(\s)(\d{2}:\d{2}:\d{2})(\s)(\d{4})/$5$3$9:$7/',1,longDate);
```

File 1.txt marcin.sosnowski Fri OCT 12 04:56:33 2016

```
newDate = input(Date,datetime18.);
```

LB data imputaion

LBSTRESC='1.1’;
LBSTRESC='.1';
LBSTRESC='-1.1' ;
LBSTRESC='-.1';
LBSTRESC='00.1’;
LBSTRESC='1.';
LBSTRESC='1';

LBSTRESN=input(LBSTRESC,best.);

LBSTRESC='1,1’;
tranwrd(LBSTRESC, ",", ".");

LBSTRESC='1..1’;
tranwrd(LBSTRESC, ". .", ".");

LBSTRESN= input(prxchange('s/(-?\d*)([,.]{1,})(\d*)/$1.$3/',1,LBSTRESC),best.);

LBSTRESC='>1.1';
LBSTRESC='>=1.1';
LBSTRESC='>=1..1'

LBSTRESN= input(prxchange('s/(?:[><=]*)(-?\d*)([,.]{1,})(\d*)/$1.$3/',1,LBSTRESC),best.);

Sometimes regular expression are more programmer friendly

```
LBSTRESN=prxchange('s(.*)(-)(.*)/$3/',1, LBSTRESC);

Dir=prxchange('s/(WOV)(\w+)/$2/',1,DTYPE);

AETOXGR = prxchange('s/GR(ADE)? *(\d)/$2/i',1,AETOXGR);
```

„Dynamic” expressions

```
if not missing(LSLOCD) then prx='s#^\Q'||strip(LSLOCD)||'\E$##';
else prx='s###';
ZILOC = upcase(catx(' - ',ZILOC,catx(', ',LSLOCD,prxchange(prx,1,strip(LSLOCOS)) )));
```

Output's

Remove special characters

AETERM=prxchange('s/[^\x20-\x7F]/ /',-1,AETERM);

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

128	Ç	144	É	160	á	176	▒	192	└	208	╨	224	α	240	≡			
129	ü	145	æ	161	í	177	▒	193	┴	209	╤	225	ß	241	±			
130	é	146	Æ	162	ó	178	▓	194	┬	210	╥	226	Γ	242	≥			
131	â	147	ô	163	ú	179	|	195	├	211	╙	227	π	243	≤			
132	ä	148	ö	164	ñ	180	┤	196	─	212	╘	228	Σ	244	⌠			
133	à	149	ò	165	Ñ	181	╡	197	┼	213	╒	229	σ	245	⌡			
134	å	150	û	166	ª	182	╢	198	╞	214	╓	230	µ	246	÷			
135	ç	151	ù	167	º	183	╖	199	╟	215	╫	231	τ	247	≈			
136	ê	152	ÿ	168	¿	184	╕	200	╚	216	╪	232	Φ	248	°			
137	ë	153	Ö	169	⌐	185	╣	201	╔	217	┘	233	Θ	249	·			
138	è	154	Ü	170	¬	186	╢	202	╩	218	┌	234	Ω	250	·			
139	ï	155	¢	171	½	187	╗	203	╦	219	▉	235	δ	251	√			
140	î	156	£	172	¼	188	╝	204	╠	220	▄	236	∞	252	n			
141	ì	157	¥	173	¡	189	╜	205	═	221	▌	237	φ	253	²			
142	Ä	158	₧	174	«	190	╛	206	╬	222	▐	238	ε	254	■			
143	Å	159	ƒ	175	»	191	┐	207	╧	223	▀	239	∩	255				

Source: www.LookupTables.com

Output’s

newAETERM = prxchange('s/(.{1,22})(\b)/$1@$2/',-1,strip(AETERM);

ID	Adverse Event
01	Abdominal discomfort
02	Abdominal hernia
03	Adjustment disorder wi th depressed mood
04	Aspartate aminotransfe rase increased
05	Blood thyroid stimulat ing hormone increased
06	Viral upper respirator y tract infection

ID	Adverse Event
01	Abdominal discomfort
02	Abdominal hernia
03	Adjustment disorder with depressed mood
04	Aspartate aminotransferase increased
05	Blood thyroid stimulating hormone increased
06	Viral upper respiratory tract infection

Some fun with numbers

col='1(1.1)'
col='65(100)'
col='0'
col='9 (99.9)'

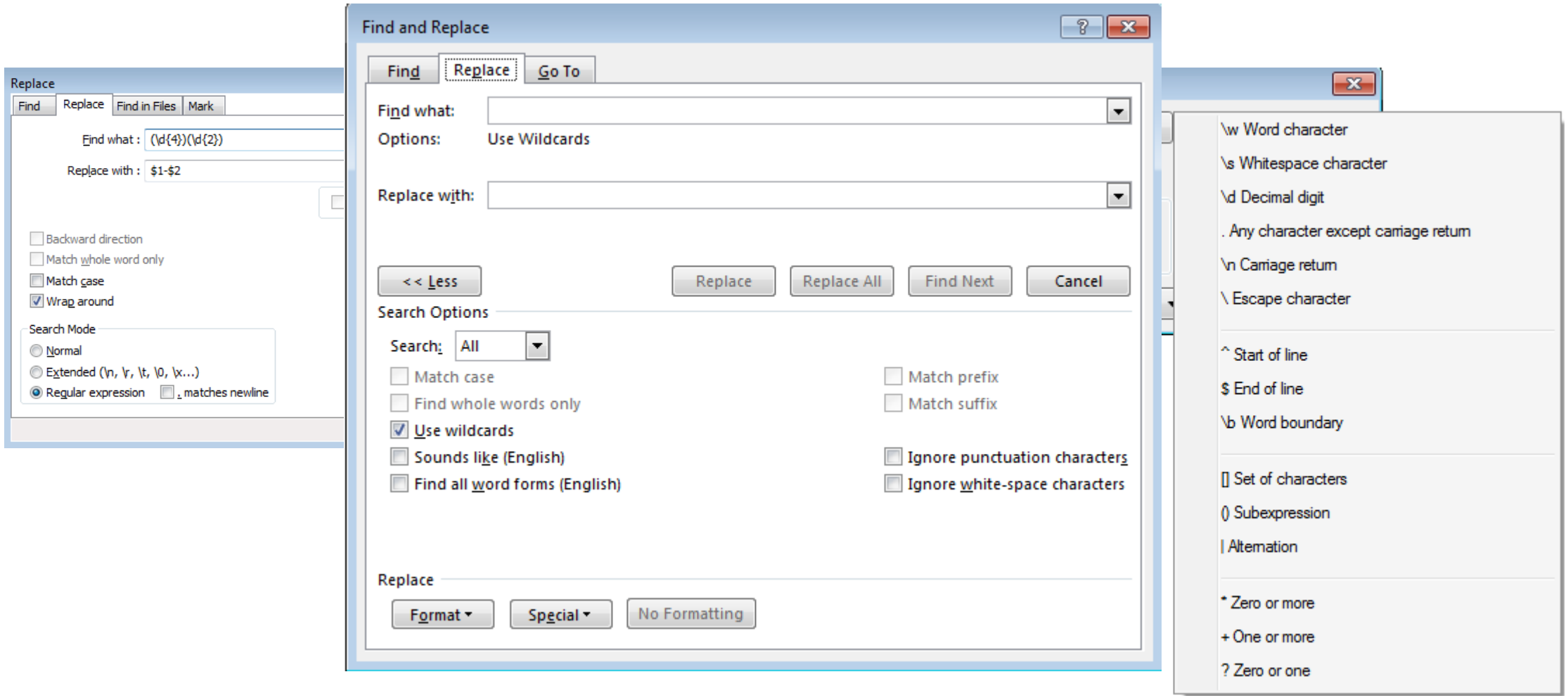
Remove percentages
newCol=prxchange('s/(\d+)(.*)/$1/',1,strip(col));

Keep only percentages
newCol = prxchange('s/(\d+)((\()(.*)(\)))?/$4/',1,strip(col))

Proc format(9.3)

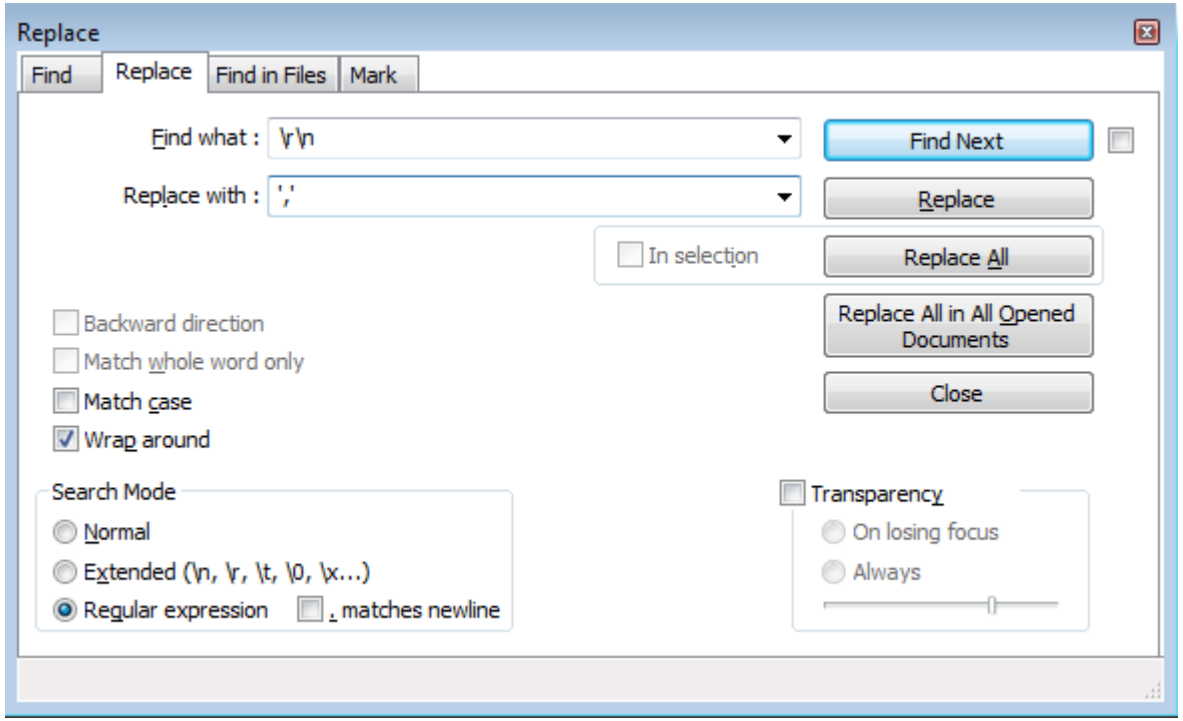
```
proc format;
  invalue xxx (default=20)
    '/(\d+):(\d\d)(?:\.(\d+))?/' (REGEXP) = [time8.]
  ;
run;
```

Editors


Replace
Find
Replace
Find in Files
Mark
Find what : (\d{4})(\d{2})
Replace with : $1-$2
Backward direction
Match whole word only
Match case
Wrap around
Search Mode
Normal
Extended (\n, \r, \t, \0, \x...)
Regular expression
. matches newline
Find and Replace
Find
Replace
Go To
Find what:
Options: Use Wildcards
Replace with:
<< Less
Replace
Replace All
Find Next
Cancel
Search Options
Search: All
Match case
Find whole words only
Use wildcards
Sounds like (English)
Find all word forms (English)
Match prefix
Match suffix
Ignore punctuation characters
Ignore white-space characters
Replace
Format
Special
No Formatting
\w Word character
\s Whitespace character
\d Decimal digit
. Any character except carriage return
\n Carriage return
\ Escape character
^ Start of line
$ End of line
\b Word boundary
[] Set of characters
() Subexpression
| Alternation
* Zero or more
+ One or more
? Zero or one

Other

```
ID01
ID02
ID03
ID04
ID05
ID06
ID07
ID08
ID09
```




```
ID01','ID02','ID03','ID04','ID05','ID06','ID07','ID08','ID09
```

Other

Egrep –l „(sdtmLib\.)(SUPP)?(\w{2})” *.sas

Pinnacle21

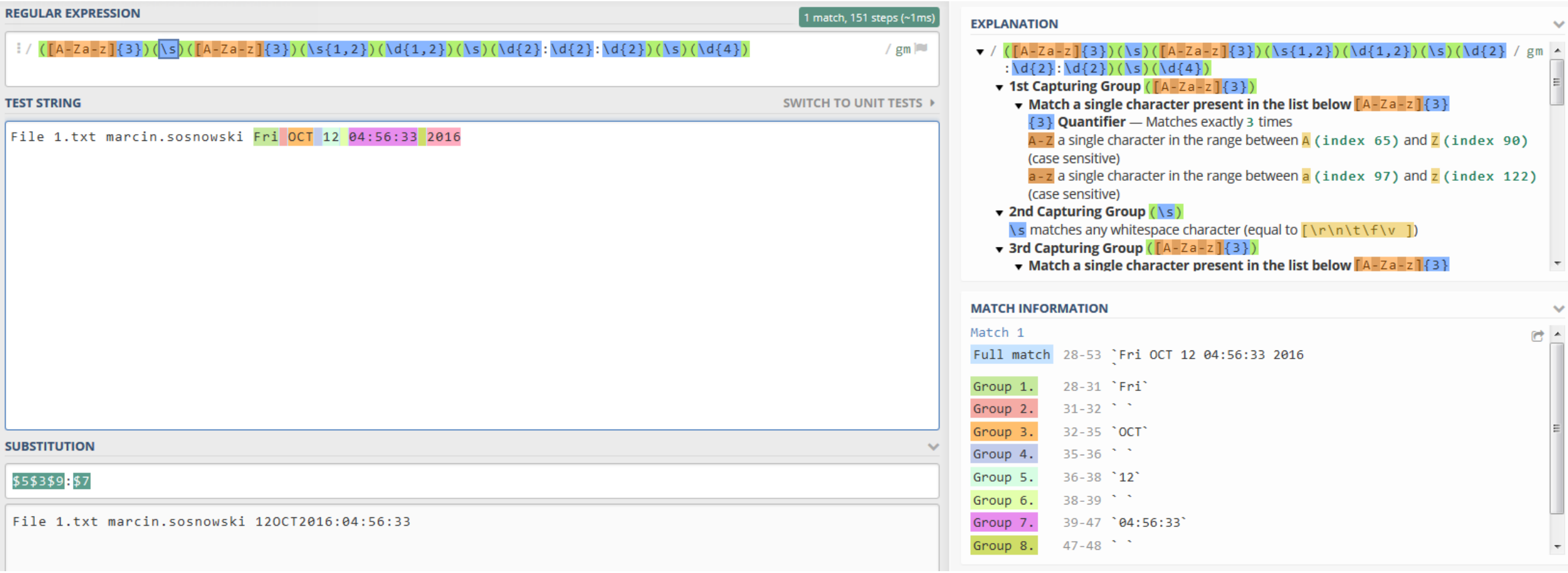
3.3 Regular Expression Rule

The Regular Expression, or Regex, Rule allows a variable's value to be validated against a pre-defined pattern. For example, you are able to enforce the length of a variable's value, and specify that it must be alpha-numeric, etc. Regular expressions can be quite complex, however, and how to define them is beyond the scope of this document. However, there are many online resources available that can help explain how regular expressions should be written. The Regular Expression Rule introduces the Test attribute, which contains the regular expression string used to validate the data. If the data does not match the regular expression, then the record will fail that Validation Rule. The structure of the Regex Rule:

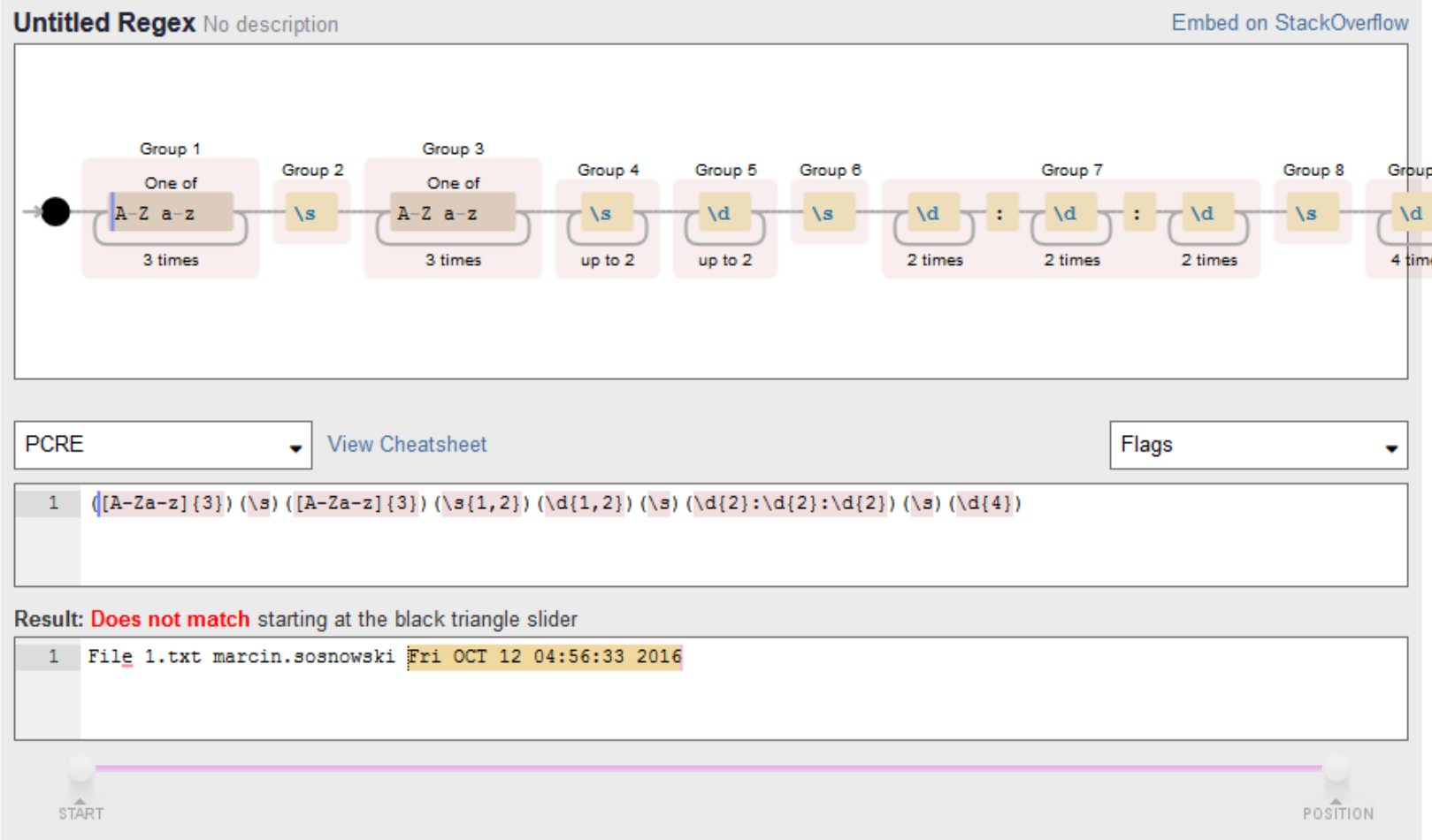
```
<val:Regex ID="id" Variable="variable" Test="regular expression"
        Message="message"
        Description="description"
        Type="type" Severity="severity" Warn="warn"/>
```

<val:Regex ID="**SD0017**" PublisherID="**FDAC057**" Message="**Invalid value for %Domain%TEST variable**"
Description="**The value of Name of Measurement, Test or Examination (--TEST) should be no more than 40 characters in length.**"
Category="**Format„**
Type="**Error**"
Variable="**%Domain%TEST„**
Test="**.{0,40}**"/>

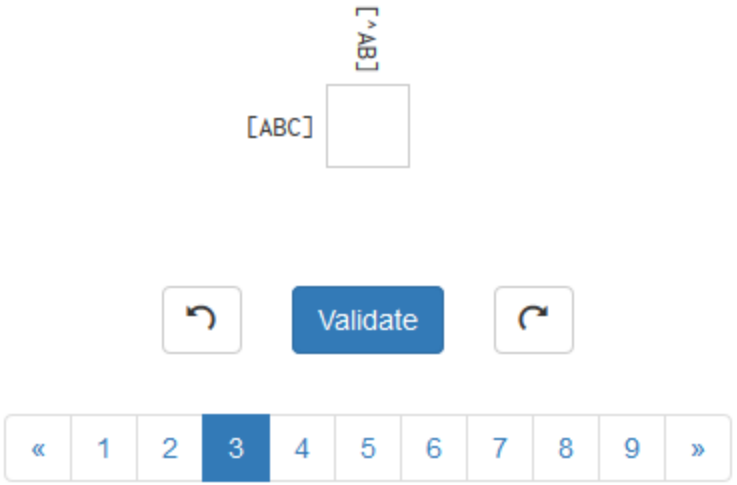
https://regex101.com/



https://www.debuggex.com/



https://regexcrossword.com/



Questions?

Thank you!

Marcin.Sosnowski@Statfinn.com