SAS9 – Perl Regular Expressions Tip Sheet

Functions and Call Routines

regex-id **= prxparse(*perl-regex*)**
Compile Perl regular expression ***perl-regex*** and return ***regex-id*** to be used by other PRX functions.

pos **= prxmatch(*regex-id* | *perl-regex*, *source*)**
Search in ***source*** and return position of match or zero if no match is found.

new-string **= prxchange(*regex-id* | *perl-regex*, *times*, *old-string*)**
Search and replace ***times*** number of times in ***old-string*** and return modified string in ***new-string***.

call prxchange(*regex-id*, *times*, *old-string*, *new-string*, *res-length*, *trunc-value*, *num-of-changes*)
Same as prior example and place length of result in ***res-length***, if result is too long to fit into ***new-string***, ***trunc-value*** is set to 1, and the number of changes is placed in ***num-of-changes***.

text **= prxposn(*regex-id*, *n*, *source*)**
After a call to **prxmatch** or **prxchange**, **prxposn** return the text of capture buffer ***n***.

call prxposn(*regex-id*, *n*, *pos*, *len*)
After a call to **prxmatch** or **prxchange**, **call prxposn** sets ***pos*** and ***len*** to the position and length of capture buffer ***n***.

call prxnext(*regex-id*, *start*, *stop*, *source*, *pos*, *len*)
Search in ***source*** between positions ***start*** and ***stop***. Set ***pos*** and ***len*** to the position and length of the match. Also set ***start*** to ***pos***+***len***+1 so another search can easily begin where this one left off.

call prxdebug(*on-off*)
Pass 1 to enable debug output to the SAS Log.
Pass 0 to disable debug output to the SAS Log.

call prxfree(*regex-id*)
Free memory for a ***regex-id*** returned by **prxparse**.

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.

SAS9 – Perl Regular Expressions Tip Sheet

Basic Example

```
data _null_;
  pos=prxmatch('/world/',
               'Hello world!');
  put pos=;

  txt=prxchange('s/world/planet/',
                -1, 'Hello world!');
  put txt=;
run;
```

Output:

```
   pos=7;
   txt=Hello planet;
```

Data Validation

```
data phone_numbers;
  length first last phone $ 16;
  input first last phone & $16.;
datalines;
Thomas Archer     (919)319-1677
Lucy Barr         800-899-2164
Tom Joad          (508) 852-2146
Laurie Gil        (252)152-7583
;

data invalid;
  set phone_numbers;
  where not
  prxmatch("/\([2-9]\d\d\) ?" ||
           "[2-9]\d\d-\d\d\d\d/",phone);
run;

proc sql;  /* Same as prior data step */
  create table invalid as
  select * from phone_numbers
  where not
  prxmatch("/\([2-9]\d\d\) ?" ||
           "[2-9]\d\d-\d\d\d\d/",phone);
quit;
```

Output:

```
   Obs    first     last        phone
    1     Lucy      Barr    800-899-2164
    2     Laurie    Gil     (252)152-7583
```

Search and Replace #1

```
data _null_;
   input;
   _infile_ =
     prxchange('s/</&lt;/', -1, _infile_);
   put _infile_;
datalines;
x + y < 15
x < 10 < y
y < 11
;
```

Output:

```
   x + y &lt; 15
   x &lt; 10 &lt; y
   y &lt; 11
```

Search and Replace #2

```
data reversed_names;
   input name & $32.;
datalines;
Jones, Fred
Kavich, Kate
Turley, Ron
Dulix, Yolanda
;

data names;
  set reversed_names;
  name = prxchange('s/(\w+), (\w+)/$2 $1/',
                   -1, name);
run;

proc sql;  /* Same as prior data step */
   create table names as
   select
     prxchange('s/(\w+), (\w+)/$2 $1/',
               -1, name)
   as name
   from reversed_names;
quit;
```

Output:

```
   Obs        name
    1     Fred Jones
    2     Kate Kavich
    3     Ron Turley
    4     Yolanda Dulix
```

Search and Extract

```
data _null_;
  length first last phone $ 16;
  retain re;
  if _N_ = 1 then do;
   re = prxparse("/\(([2-9]\d\d)\) ?" ||
                 "[2-9]\d\d-\d\d\d\d/");
  end;

  input first last phone & $16.;

  if prxmatch(re, phone) then do;
    area_code = prxposn(re, 1, phone);
    if area_code ^in ("828" "336"
                      "704" "910"
                      "919" "252") then
    putlog "NOTE: Not in NC: "
           first last phone;
  end;
datalines;
Thomas Archer     (919)319-1677
Lucy Barr         (800)899-2164
Tom Joad          (508) 852-2146
Laurie Gil        (252)352-7583
;
```

Output:

```
   NOTE: Not in NC, Lucy Barr (800)899-2164
   NOTE: Not in NC, Tom Joad (508) 852-2146
```

For complete information refer to the
Base SAS 9.1.3 documentation at
http://support.sas.com/v9doc