Subject: Re: stregex - lookaround operators?
Posted by rtowler on Thu, 18 Oct 2007 16:53:57 GMT
View Forum Message <> Reply to Message

Thanks Chris,

I ended up pretty much doing the same thing you did but I used
STRSPLIT or STRMID depending on my mood.

```
rid= STREGEX(dg.dgram, '[0-9]*</rid>', /EXTRACT)
rid = LONG((STRSPLIT(rid, '<'))[0])

xcvrs = STREGEX(dg.dgram, '[ a-zA-Z0-9,\-]*</value>', /EXTRACT)
xcvrs = STRMID(xcvrs, 0, STRLEN(xcvrs) - 8)
```

Luckily my input strings will always only contain one pair of tags for
each type so I don't have to resort to loops for processing. Still, I
think this is rather ugly and will request that the regex engine in
IDL be updated to include the lookaround operators.

-Rick

On Oct 18, 12:56 pm, Spon wrote:
```
>  On Oct 17, 5:35 pm,  wrote:
>
>>  Does stregex support the lookahead and lookbehind operators?  I'm
>>  guessing it doesn't, which is a real bummer.  Specifically I am trying
>>  to use the lookahead operator to extract data from an XML like string:
>>  '[^>]*(?=</rid>)' which should extract all characters before the </
>>  rid> tag up to ">".  But I get an error:
>
>>  STREGEX: Error processing regular expression: [^>]*(?=</rid>)
>>           repetition-operator operand invalid
>
>>  I think the expression is valid.  Has anyone used the lookaround
>>  operators in IDL?
>
>>  -r
>
>  This is the best I could do. Not elegant, probably not efficient, but
>  it throws up a few questions of its own:
>
>  *******
>
>  FUNCTION STREGEXTEST, StrExpr
>
>  IF N_PARAMS() EQ 0 THEN $
>   StrExpr = ['<rid> Stuff in here </rid>', $
>   '<rid> All of this stuff in here </rid>', $
>   '<html> Not this stuff </html>', 'frog', $
>   '>.<', '<grid> Not this either </grid>', $
>   '</html> <rid> This stuff too </rid>', $
>   '<rid> Even this</rid> <foo>!</foo>', $
>   'The acid test </rid>' ]
>
>   ; StrExprTemp = StrExpr
>
>  RegStr = '>[^>]*(</rid>)'
>
>  Streg = STREGEX (StrExpr, RegStr, $
>   LENGTH = Length)
>  ExtrStreg = STREGEX (StrExpr, RegStr, /EXTR)
>
>  PRINT, Streg
>  PRINT, ExtrStreg
>
>  Index = WHERE (Streg GE 0, Count)
>  IF Count EQ 0 THEN RETURN, ''
>
>  Streg  =  Streg   [Index]
>  Length =  Length  [Index]
>  PosExp =  StrExpr [Index]
>
>  Streg  += 1 ; Manually move your location
>  Length -= 7 ; and length pointers to
>              ; exclude the Lookarounds
>
>   ;  ; This doesn't work for string arrays,
>   ;  ; even if Streg and Length have the
>   ;  ; same dimensions:
>   ; Strings = STRMID (PosExp, Streg, Length)
>
>  NS = N_ELEMENTS (Streg)
>  Strings = STRARR (NS)
>  FOR i = 0L, NS - 1 DO $
>   Strings [i] = STRMID (PosExp [i], $
>    Streg [i], Length [i] )
>
>  HELP,   Strings
>  PRINT,  Strings
>  RETURN, Strings
>  END
>
>  ********
>
```

> As you can see, I had foolishly assumed that if you call STRMID with
> three arrays of equal dimensions, IDL would somehow know to
> 'vectorise' the calculation, but I had to pump it through a FOR loop
> instead. Can someone show me a way around this?
>
> The other commented-out line is this:
> ; StrExprTemp = StrExpr
> Which I was using because I thought STREGEX (or possibly STRMID) was
> changing my input function (I've written functions of my own that
> ended up doing this, and watched in dismay as my nice 4-dimensional
> array was decimated to a tiny subset, when all I was doing was trying
> to was extract the subset from the data, leaving the original intact.
> It's easy not to do a second time, but quite frustrating the first
> time you do it!)
> I seemed to be getting a large array of blank strings after using
> STREGEX and them STRMID on the input array (the resulting array was of
> expected dimensions, just empty, contrary to my expectations.) I can't
> seem to replicate this now, so chances are it was some silly error on
> my part, but I thought I'd ask if anyone else had come across this?
>
> Thanks,
> Chris