Subject: A case for lookarounds in StRegEx()
Posted by Matthew Argall on Fri, 27 Jun 2014 00:53:59 GMT

View Forum Message <> Reply to Message

I want to make a case for the stregex function to recognize lookarounds.

Say I have a list of tokens YMd. The tokens are identifiable because they are preceded by %. The "%" character can be escaped by "\". Try to extract the tokens following %.

The following case is successful. There are three tokens I want to find, so I search for "%" followed by any one of the three characters "[YMd]" and extract it with "()", then eat up any extra characters that are not % with "[^%]*".

```
IDL> print, stregex('file_%Y%M%d.txt', strjoin(replicate('%([YMd])[^%]*', 3)), /SUBEXP,
/EXTRACT)
%Y%M%d.txt Y M d
```

Now I want to change the "%Y" character to "\%Y" so that the % is escaped and Y is excluded from the search. The following successfully skips "\%Y" and finds "%M", but fails to find "%d" because the "%" character that precedes "d" has been eaten up by a search for "[^\]" -- i.e. "[^\]" is of length one, whereas a negative lookbehind is of length zero.

```
IDL> print, stregex('file_\%Y%M%d.txt', strjoin(replicate('%([YMd])[^%]*', 3)), /SUBEXP,
/EXTRACT)
```

```
IDL> print, stregex('file_\%Y%M%d.txt', strjoin(replicate('[^\]%([YMd])[^%]*', 1)), /SUBEXP,
/EXTRACT)
Y%M M
```

Using the Python negative lookbehind notation "(?<!\\)%[YMd]" avoids %Y and matches %M and %d successfully (test here: https://www.debuggex.com/)

This is just one example of where they are useful.

TLDR; negative lookbehinds make searching for escaped characters really easy.