Yalan unter Unix (und Lob für Yalan :-)

8 November, 2005 - 23:03 — StephanFrenzel

In der folgenden Mail beschreibt "Rich" seine Erfahrung mit Yalan unter Unix. Er hat noch einigen Anpassungsbedarf identifiziert (wir werden seine Anregungen in die nächste Version einfließen lassen), was ihn aber nicht daran hindert, sich lobend zu äußern - Danke!

Many thanks for letting me use YALAN, it's proving very useful. Looking at the website and docs it seems your primarily developing it for Windows and MVS. What I need it for is Windows and Unix. For Windows, as I say, it's fine, it works well. For Unix, you'll be pleased to know, it is essentially fine too, it just require one or two small changes which you may wish to employ...

1.

The first is that SAS Unix just cannot deal with mixed case macro variables. An unbelievably absurd point but true. So macros such as 'rdsFileFinder' are just not seen by SAS. However, I've changed all the macros (see attachment) so they can be read by Unix. the rename is simple; basically macro 'rdsFileFinder' becomes 'macro rds_file_finder'. Not too tricky and obviously works fine in Windows (no idea about MVS).

(..as an aside, I noticed macro 'rdsFileFinder' was not referenced by any of your other macros though 'rds_fileFinder' was. I'm assuming a typo or old macro name. I changed the macro referencing 'rds_FileFinder' to the new 'rds_File_Finder' name. I hope that's OK..)

2.

At some point you must read in the log file (bear in mind I only use the 'rds_log_reader_sas' macro) and input using delimiters. Unix annoyingly leaves on its line endings so all of the values of the character variables from your output datasets have this line ending. Knowing that the hex value for unix line endings is "'0d'x", you could stick this in your "dlm" parameter alongside the others (e.g. "dlm='0a0d'x" would delimit on tabs *and* Unix line endings) or explicitly filter them out with the following sort of code:-

```
data IO(drop=i);
    set IO;
   array temp[*] _character_;
   do i=1 to dim(temp);
       temp[i]=translate(temp[i],'','0d'x);
   end;
run;
```

This is what I do for now.

3.

You may or may not know this but the SAS god have decided that for SAS 9, Proc SQL cannot write to a table it is reading from without printing a warning to the log (e.g. "create table x as select * from x" is no longer advised). This is not the end of the world, of course, as it still works but as I say, a warning is printed for each log read (incidentally, I *think* it is the section of code that starts "create table log1 as select logDigest..." where the problem is caused).

Anyway, I hope this helps you and well done for producing such an excellent tool.

Many thanks, Rich

oben