
Subject: A routine to annotate PS files

Posted by [MarioIncandenza](#) on Sat, 09 Jan 2010 00:44:28 GMT

[View Forum Message](#) <> [Reply to Message](#)

Happy New Year Everybody!

I asked about this last year, and someone pointed me to the DEVICE,/OUTPUT and SCOPE_TRACEBACK commands, which turned out to hit the spot.

So here is a routine you can use to personalize your PS files and make them easier to search. The most obvious application is to attach the IDL calling stack in the postscript file, so that you can determine what script exactly made what file and when. However, this procedure is extensible, and I suspect people will find more uses for it.

```
pro annotate_psfile,extra_tags=extra_tags
; This function is intended to permit the user to add annotation into
; a PostScript file using the device,/output statement,
; in comment lines: %% Tag : Value
; It's default behavior is to add a series of tags as follows:
; GenRoutine: The routine that called this function
; GenFile: The filename of the script that called this function
; CallRoutine: The routines above in the trace (may be multiple
; 'CallRoutine' tags)
; CallFiles: The filenames above in the trace (may be multiple
; 'CallFiles' tags)
; YYYYMMDDHHMM: An easily-parseable timecode (time this function was
; called)
; You can also add your own tags with the EXTRA_TAGS keyword, which
; should be a structure of (TAGNAME).TAGVALUE. For best results, all
; TAGVALUE values should be strings.
; This procedure must be called while the postscript file is open.
tagf='("%% ",a," : ",a)'; %% Tag : Value
trace=scope_traceback(/structure)
ntrace=n_elements(trace)
geni=ntrace-2 ; index to proximate calling script
if(ntrace gt 3) then calli=indgen(ntrace-3)+1
tag0=string('GenRoutine',trace[geni].routine,format=tagf)
tag1=string('GenFile',trace[geni].filename,format=tagf)
tags=[tag0,tag1]
if(ntrace gt 3) then begin
  for icall=0,ntrace-4 do begin
    tag0=string('CallRoutine',trace[calli
[icall]].routine,format=tagf)
    tag1=string('CallFile',trace[calli
[icall]].filename,format=tagf)
```

```

        tags=[tags,tag0,tag1]
    endfor
endif
timecode=string(julday(),format='(C
(CYI4.4,CMOI2.2,CDI2.2,CHI2.2,CMI2.2))')
tag0=string('TimeCode: ',timecode,format=tagf)
tags=[tags,tag0]
if n_elements(extra_tags) ne 0 then begin
    extranames=tag_names(extra_tags)
    for iextra=0l,n_elements(extranames)-1 do begin
        tag0=string(extranames[iextra],extra_tags.(iextra),format=tagf)
        tags=[tags,tag0]
    endfor
endif
print,tags,format='(A)'
ntags=n_elements(tags)
for itag=0l,ntags-1 do device,output=tags[itag]
return
end

```
