
Subject: HELP! Annoying IDL glitches...

Posted by [deb](#) on Thu, 16 May 1996 07:00:00 GMT

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

Here are some questions I'm hoping seasoned IDL users will be able to help me out with.

1. Why does STRMID always give a non-fatal error announcing that it cannot convert the given string to type Long? The function itself appears to extract the string expression properly.
2. Why does XLOADCT update color tables in real time on the Mac but not on the PC?
3. Why does XYOUTS, when called with the keyword TEXT_AXES = 3 rotate text such that it's mirror imaged of what one would expect for a regular y-axis label?
4. What's the EOF marker that IDL searches for and is it different on the Mac vs the PC? Sometimes when reading in an array of signed data i get an EOF message long before the true EOF is reached. Is there a trick to reading in signed vs unsigned data? On a related note, why would a formatted read statement (reading string data) say it reached the end of a line too early, shouldn't it just pad the variable with blanks? Is there a way to read say 3 variables off a single line when one really only knows the format of the first two? ie, instead of readf,unit,format='(a34,F5.3,a80)',a,b,c, can one leave off that a80 and still expect to read 3 variables off the line (with the third variable being a string of arbitrary length)?
- 4a. Along similar lines, why is it that on te Mac i can set graphics preferences to 256 colors and it works whereas on the PC i set the preferences to 256 colors and !D.n_colors still comes up with many more?
- 4b. SPEaking of Mac vs PC versions of IDL, what other undocumented annoying differences have people noticed?
5. Can one open a single (large) graphics window with scroll bars? (Not the same as the SLIDE_IMAGE command, but along the lines of the scroll window which is incorporated into that)
6. Suppose one opens a datafile which contains an ascii header and then

a bunch of binary junk which ends in an ascii-readable tag. Following the header is a bunch of binary data. I'd like to piece thru the file until i find the final ascii-readable tag which denotes the start of the actual data. This should be simple enough, and when the binary junk between the ascii-header and the final ascii tag is stripped out things seem to work ok. When the binary junk is not stripped out, sometimes a STRPOS command finds the tag but frequently IDL chokes; when using a simple readf command, the error message says the input line is too long for the input buffer, even though it really isn't anywhere near 32767 characters long. Why? How does one get around this? Is there a way to read in ascii string data until i find the tag which preceeds the binary junk, then switch to reading in a binary string and looking for this ascii-readable final tag within that string and then positioning the file pointer right after that so i can extract my data? Does any of this make any sense whatsoever? (I didn't write the app that spits out the datafiles and the guy that did isn't talking, at least not enough to be of much use)

Has anyone else run into similar problems and how did you fix them? ANY other pointers (no pun intended) on file i/o and/or Mac vs PC things to watch out for?

Many thanks,

-Deb Summa
summa@lanl.gov