Subject: New IDL features related to recent newsgroup discussions
Posted by info on Fri, 30 Apr 1993 21:19:53 GMT
View Forum Message <> Reply to Message

There have a number of postings recently regarding "wrapper" routines, the EXECUTE function and annotating plots with date axes. The following is presented in order to avoid moot discussions.

DATE AXES

Ken Deere of Naval Research Lab asked about making plots with date axes. This is easy to do with the [XYZ]TICKFORMAT keyword which appeared in IDL Version 3.0 and the LABEL_DATE function which is included at the end of this message. The [XYZ]TICKFORMAT keyword allows users to provide their own axis annotation functions, filling a wide range of requirements.

The following improvements have been made to IDL. They will appear in Version 3.1, due around the first of June.

KEYWORD INHERITANCE

Keyword inheritance allows IDL routines to pass keyword parameters not defined in their declaration on to routines they call. This greatly simplifies writing "wrapper" routines, which are variations of a system or user-provided routine.

If a routine is defined with a formal keyword parameter named _EXTRA, pairs of unrecognized keywords and values are placed in an anonymous structure. The name of each unrecognized keyword becomes the tag name, and its value is the tag value. Anonymous structures containing name/value pairs may be manipulated with the CREATE_STRUCT function.

When the keyword _EXTRA appears in a procedure or function call, its argument is a structure containing additional keyword/value pairs which are inserted into the argument list. The value of _EXTRA will be "undefined", if no additional keyword parameters are present.

For example, the procedure definition:

```
PRO Test, a, b, _EXTRA = e, COLOR = color
```

places unrecognized keywords and their values into the variable e. If there are no unrecognized keywords, e will be undefined.

The call:

Test, x, y, COLOR=3, LINESTYLE = 4, THICK=5
results in variable "e" within TEST, containing an anonymous structure with a value of { LINESTYLE: 4, THICK: 5}.

These keyword/value pairs are then passed from TEST to the next routine, PLOT in this example, with the _EXTRA keyword:
PLOT, a, b, COLOR = color, _EXTRA = e

Keywords passed into a routine override previous settings of that keyword. For example, the call:
PLOT, a, b, COLOR = color, _EXTRA = { COLOR: 12}
specifies a color index of 12 to PLOT.

CREATE_STRUCT FUNCTION

The CREATE_STRUCT function has been added to create, concatenate, and remove fields from anonymous structures. CREATE_STRUCT, creates an anonymous structure given structures and/or pairs of tagnames and values. Tagname parameters may be either scalar strings or string arrays.

Some examples follow:
p = CREATE_STRUCT('A', 1, 'B', 'xxx')
creates an anonymous structure: { A: 1, B: 'xxx'}.
p = CREATE_STRUCT('FIRST', 0, p, 'LAST', 3)
adds the tags FIRST and LAST to p, making: p contain { FIRST: 0, A: 1, B: 'xxx', LAST: 3}. The statement:
p = CREATE_STRUCT(['A','B','C'], 1, 2, 3)
creates the structure { A: 1, B: 2, C: 3}. Finally, tag/value pairs may be removed via the REMOVE keyword:
p = CREATE_STRUCT(p, 'LAST', 10, REMOVE=[0,1])
returns { C:3, LAST:10}, by removing tag/values 0 and 1 from p, and then adding the LAST field.

IMPROVED EXECUTE FUNCTION

The EXECUTE function now allows nested calls to EXECUTE, removing the previous restriction that EXECUTE could not call a routine that used EXECUTE.

A bug, reported by Jeff de la Beaujardiere of the University of Hawaii, in Version 3.0 EXECUTE that caused the Code Area to become full after numerous repeated EXECUTES from the single statement mode has been fixed.

```
 ;--------------LABEL_DATE---------------------------------- ----------------
FUNCTION LABEL_DATE, axis, index, x, DATE_FORMAT = format, MONTHS = months
;+
; NAME:
; LABEL_DATE
;
; PURPOSE:
; A function to label axes with dates.
; CATEGORY:
; Plotting.
; CALLING SEQUENCE:
; To set up:
;  dummy = LABEL_DATE(DATE_FORMAT='string')
; Then to use:
;  PLOT, x, y, XTICKFORMAT='LABEL_DATE'
; INPUTS:
; No explicit user defined inputs.  When called from the plotting
; routines, the input parameters are (Axis, Index, Value)
; KEYWORD PARAMETERS:
; DATE_FORMAT = a format string which may contain the following:
;  %M for month (3 character abbr)
;  %N for month (2 digit abbr)
;  %D for day of month,
;  %Y for 4 digit year.
;  %Z for last two digits of year.
;  %% is %.
;  Other characters are passed directly thru.
;  For example, '%M %D, %Y' prints DEC 11, 1993
;   '%M %2Y' yields DEC 93
;   '%D-%M' yields 11-DEC
;   '%D/%N/%Y' yields 11/12/1993
;   '%M!C%Y' yields DEC on the top line, 1993 on
;   the bottom (!C is the new line graphic command).
; MONTHS = The names of the months, a twelve element string array.
;  If omitted, use Jan, Feb, ..., Dec.
; OUTPUTS:
; The date string which is then plotted.
; COMMON BLOCKS:
; LABEL_DATE_COM.
; SIDE EFFECTS:
; None.
; RESTRICTIONS:
; Only one date axis may be simultaneously active.
```

```
; PROCEDURE:
; Straightforward.
; EXAMPLE:
; For example, to plot from Jan 1, 1993, to July 12, 1994:
; Start_date = julday(1, 1, 1993)
; End_date = julday(7, 12, 1994)
; Dummy = LABEL_DATE(DATE_FORMAT='%N/%D')  ;Simple mm/dd
; x = findgen(end_date+1 - start_date) + start_date ;Time axis
; PLOT, x, sqrt(x), XTICKFORMAT = 'LABEL_DATE', XSTYLE=1
; (Plot with X axis style set to exact.)
;
; MODIFICATION HISTORY:
; DMS, RSI. April, 1993. Written.
;-

COMMON label_date_com, fmt, month_chr

if keyword_set(format) then begin ;Save format string?
 if keyword_set(months) then month_chr = months $
 else month_chr = ['Jan','Feb','Mar', 'Apr', 'May', 'Jun', 'Jul', $
  'Aug', 'Sep', 'Oct', 'Nov', 'Dec']
 fmt = format
 return, 0
 endif

if n_elements(month_chr) ne 12 or n_elements(fmt) le 0 then $
 message,' Not initialized.'

caldat, long(x), month, day, year ;Get the calendar date from julian
n = strlen(fmt)
out = ''

for i=0, n-1 do begin   ;Each format character...
    c = strmid(fmt, i, 1)  ;The character.
    if c eq '%' then begin
 i = i + 1
 c = strmid(fmt, i, 1)  ;The function
 case c of  ;format character?
 'M' : out = out + month_chr(month-1)
 'N' : out = out + string(format='(i2.2)', month)
 'D' : out = out + string(format='(i2.2)', day)
 'Y' : out = out + string(format='(i4)', year)
 'Z' : out = out + string(format='(i2.2)', year  mod 100)
 '%' : out = out + '%'
 else : message, 'Illegal character in date format string: '+fmt
 endcase
    endif else out = out + c
endfor
```

```
return, out
end
```