Subject: Re: xyouts with p.multi
Posted by Paul van Delst on Fri, 25 May 2001 13:32:59 GMT
View Forum Message <> Reply to Message

"Christopher W. O'Dell" wrote:
>
> I am plotting six graphs on one page using !p.multi =
> [0,2,3]
> I am trying to put some text in the same place on each
> graph, by using
>
> xyouts, x, y, 'text', /norm
>
> after each graph is plotted. I would think, using /norm,
> that it would put the
> text *in that graph*; but it doesn't, it puts it in some
> crazy place. Does anyone know how
> to make xyouts position the text with respect to the most
> recently plotted graph in the current window?

I smacked together a little procedure to allow me to do exactly that but for legends. It's probably already been done, and done better (particularly in the braindead way I handle the symbols). DF's example uses stuff I've never tried before (the !X.WINDOW stuff) so most of the position calculation gymnastics may be unnecessary. Anyway, the code is attached to this post (mylegend.pro). Use it, change it, wear it out.

cheers,

paulv

--
Paul van Delst A little learning is a dangerous thing;
CIMSS @ NOAA/NCEP Drink deep, or taste not the Pierian spring;
Ph: (301)763-8000 x7274 There shallow draughts intoxicate the brain,
Fax:(301)763-8545 And drinking largely sobers us again.
Alexander Pope.

```
;+
;
; NAME:
;       mylegend
;
; PURPOSE:
;       Procedure to output a plot legend in "PLOT NORMAL" coordinates. Plot normal
;       coordinates are defined as coordinates normalised for the plotting region
;       bounded by the plot axes, e.g. the bottom left axex vertex corresponds to
```

```
;       (0,0) and the top right one corresponds to (1,1).
;
; CATEGORY:
;       Graphics: Direct
;
; LANGUAGE:
;       IDL v5.4
;
; CALLING SEQUENCE:
;       mylegend, x_legend_pos,         $
;               y_legend_pos,         $
;               legend_text,          $
;               dx_pos    = dx_pos,    $
;               color     = color,     $
;               psym      = psym,      $
;               symsize   = symsize,   $
;               linestyle = linestyle, $
;               charsize  = charsize,  $
;               _extra    = extra
;
; INPUTS:
;       x_legend_pos:   X-coordinate of the legend start position in "PLOT NORMAL"
;                       coordinates. Plot normal coordinates are coordinates normalised
;                       for the plotting region bounded by the plot axes.
;
;       y_legend_pos:   Y-coordinate of the legend start position in "PLOT NORMAL"
;                       coordinates. Plot normal coordinates are coordinates normalised
;                       for the plotting region bounded by the plot axes.
;
;       legend_text:    String array containing the legend text. No zero-length
;                       elements are permitted.
;
; INPUT KEYWORD PARAMETERS:
;       dx_pos:         Length of the line in the legend, in normalised units.
;
;       color:          Array containing the color of the legend text entries.
;                       If defined, must be the same size as the LEGEND_TEXT argument.
;
;       psym:           Array containing the symbol code of the legend text entries.
;                       If defined, must be the same size as the LEGEND_TEXT argument.
;
;       symsize:        Array containing the symbol size of the legend text entries.
;                       If defined, must be the same size as the LEGEND_TEXT argument.
;
;       linestyle:      Array containing the linestyel of the legend text entries.
;                       If defined, must be the same size as the LEGEND_TEXT argument.
;
;       charsize:       Character size to use in output of the legend text.
```

```
;
;       _extra:         Any other keywords accepted by both the PLOTS and XYOUTS
;                       commands.
;
; OUTPUTS:
;       None.
;
; OUTPUT KEYWORD PARAMETERS:
;       None.
;
; CALLS:
;       None.
;
; COMMON BLOCKS:
;       None.
;
; INCLUDE FILES:
;       None.
;
; SIDE EFFECTS:
;       None.
;
; RESTRICTIONS:
;       This procedure assumes that the plotting range (transformation?)
;       has already been established by a call to PLOT. You'll still get
;       output if you haven't but if it's not where you expect it, too bad.
;
; EXAMPLE:
;       To plot a legend in the centre of the plot,
;
;         mylegend, 0.5, 0.5, $
;                 [ 'First one', 'Second one' ]
;
; CREATION HISTORY:
;       Written by:     Paul van Delst, CIMSS/SSEC, 04-Apr-2001
;                       paul.vandelst@ssec.wisc.edu
;
;  Copyright (C) 2001 Paul van Delst, CIMSS/SSEC/UW-Madison
;
;  This program is free software; you can redistribute it and/or
;  modify it under the terms of the GNU General Public License
;  as published by the Free Software Foundation; either version 2
;  of the License, or (at your option) any later version.
;
;  This program is distributed in the hope that it will be useful,
;  but WITHOUT ANY WARRANTY; without even the implied warranty of
;  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
;  GNU General Public License for more details.
```

```
;
;  You should have received a copy of the GNU General Public License
;  along with this program; if not, write to the Free Software
;  Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA  02111-1307, USA.
;
;-

PRO mylegend, x_legend_pos,         $
              y_legend_pos,         $
              legend_text,          $
              dx_pos    = dx_pos,   $
              color     = color,    $
              psym      = psym,     $
              symsize   = symsize,  $
              linestyle = linestyle, $
              charsize  = charsize,  $
              _extra    = extra



  ;-------------------------------------------------------- -------------------
  ;                     -- Set up error handler --
  ;-------------------------------------------------------- -------------------

  CATCH, error_status

  IF ( error_status NE 0 ) THEN BEGIN
    CATCH, /CANCEL
    MESSAGE, !ERROR_STATE.MSG, /CONTINUE
    HELP, /TRACEBACK
    RETURN
  ENDIF



  ;-------------------------------------------------------- -------------------
  ;                        -- Check input --
  ;-------------------------------------------------------- -------------------

  n_arguments = 3
  IF ( N_PARAMS() LT n_arguments ) THEN $
    MESSAGE, 'Invalid number of arguments.', $
             /NONAME, /NOPRINT



  ; ----------------------------------------
  ; Check that required arguments are defined
  ; ----------------------------------------
```

```
IF ( N_ELEMENTS( x_legend_pos ) EQ 0 ) THEN $
  MESSAGE, 'Input X_LEGEND_POS argument not defined!', $
         /NONAME, /NOPRINT

IF ( N_ELEMENTS( y_legend_pos ) EQ 0 ) THEN $
  MESSAGE, 'Input Y_LEGEND_POS argument not defined!', $
         /NONAME, /NOPRINT

n_entries = N_ELEMENTS( legend_text )
IF ( n_entries EQ 0 ) THEN $
  MESSAGE, 'Input LEGEND_TEXT argument not defined!', $
         /NONAME, /NOPRINT

index = WHERE( STRLEN( legend_text ) EQ 0, count )
IF ( count GT 0 ) THEN $
  MESSAGE, 'Input LEGEND_TEXT argument contains empty strings. Norty norty!', $
         /NONAME, /NOPRINT



; --------------
; Check keywords
; --------------

; -- Horizontal length of line in legend entry
IF ( N_ELEMENTS( dx_pos ) EQ 0 ) THEN dx_pos = 0.04

; -- Colours
n_colors = N_ELEMENTS( color )
IF ( n_colors EQ 0 ) THEN $
  color = MAKE_ARRAY( n_entries, VALUE = !P.COLOR ) $
ELSE $
  IF ( n_colors NE n_entries ) THEN $
    MESSAGE, 'Keyword COLOR size different from LEGEND_TEXT', $
           /NONAME, NOPRINT

; -- Symbols
n_symbols = N_ELEMENTS( psym )
IF ( n_symbols EQ 0 ) THEN $
  psym = MAKE_ARRAY( n_entries, VALUE = 0 ) $
ELSE $
  IF ( n_symbols NE n_entries ) THEN $
    MESSAGE, 'Keyword PSYM size different from LEGEND_TEXT', $
           /NONAME, NOPRINT

; -- Symbol size
n_sizes = N_ELEMENTS( symsize )
IF ( n_sizes EQ 0 ) THEN $
```

```
  symsize = MAKE_ARRAY( n_entries, VALUE = 1.0 ) $
ELSE $
  IF ( n_sizes NE n_entries ) THEN $
    MESSAGE, 'Keyword SYMSIZE size different from LEGEND_TEXT', $
            /NONAME, NOPRINT

; -- Linestyle
n_styles = N_ELEMENTS( linestyle )
IF ( n_styles EQ 0 ) THEN $
  linestyle = MAKE_ARRAY( n_entries, VALUE = 0 ) $
ELSE $
  IF ( n_styles NE n_entries ) THEN $
    MESSAGE, 'Keyword LINESTYLE size different from LEGEND_TEXT', $
            /NONAME, NOPRINT

; -- Character size
IF ( N_ELEMENTS( charsize ) EQ 0 ) THEN charsize = 0.75



 ;------------------------------------------------------------ -------------------
;            -- Get the character sizes in normal coordinates --
 ;------------------------------------------------------------ -------------------

ch_size = CONVERT_COORD( !D.X_CH_SIZE, !D.Y_CH_SIZE, /DEVICE, /TO_NORMAL )
x_ch_size = ch_size[ 0 ] * charsize
y_ch_size = ch_size[ 1 ] * charsize



 ;------------------------------------------------------------ -------------------
; -- Convert the input "PLOT NORM" coordinates to actual normal coordinates --
 ;------------------------------------------------------------ -------------------

; -- Assign axis ranges
x1 = !X.CRANGE[0]
x2 = !X.CRANGE[1]

y1 = !Y.CRANGE[0]
y2 = !Y.CRANGE[1]

; -- Apply transformations. Doesn't matter if axes are logarithmic
; -- If they are, transformation is applied to the LOG of the range.
x_data_pos = x1 + ( ( x2 - x1 ) * x_legend_pos )
x_norm_pos = !X.S[0] + ( !X.S[1] * x_data_pos )

y_data_pos = y1 + ( ( y2 - y1 ) * y_legend_pos )
y_norm_pos = !Y.S[0] + ( !Y.S[1] * y_data_pos )
```

```
 ;------------------------------------------------------------ ------------------
;                       -- Output the legend --
 ;------------------------------------------------------------ ------------------

; ------------------------------------------
; Create the array of Y legend text coordinates
;
; DELTA_CHAR defines the vertical separation of
; legend entities in Y_CH_SIZE units.
; ------------------------------------------

delta_char = 1.5
y_pos = y_norm_pos - ( FINDGEN( n_entries ) * y_ch_size * delta_char )


; ------------------------------------------------
; Loop over the legend text entries
;
; DELTA_CHAR defines the horizontal separation of
; legend entities from it's graphical descriptor
; in X_CH_SIZE units.
; ------------------------------------------------

FOR i = 0, n_entries - 1 DO BEGIN


  ; --------------------
  ; Check the symbol sign
  ; --------------------

  CASE 1 OF

    ( psym[ i ] LT 0 ): BEGIN

      ; -- Symbol and line
      PLOTS, [x_norm_pos, x_norm_pos + dx_pos], $
             [y_pos[i]  , y_pos[i]], $
             PSYM      = psym[ i ], $
             COLOR     = color[ i ], $
             SYMSIZE   = symsize[ i ], $
             LINESTYLE = linestyle[ i ], $
             /NORM, $
             _EXTRA = extra

      ; -- Legend text
```

```
  XYOUTS, x_norm_pos + dx_pos + ( delta_char * x_ch_size ), $
          y_pos[ i ] - ( y_ch_size/2.0 ), $
          legend_text[ i ], $
          COLOR    = color[ i ], $
          CHARSIZE = charsize, $
          /NORM, $
          _EXTRA = extra
END

( psym[ i ] EQ 0 ): BEGIN

  ; -- Just a line
  PLOTS, [x_norm_pos, x_norm_pos + dx_pos], $
         [y_pos[i]  , y_pos[i]], $
         COLOR     = color[ i ], $
         SYMSIZE   = symsize[ i ], $
         LINESTYLE = linestyle[ i ], $
         /NORM, $
         _EXTRA = extra

  ; -- Legend text
  XYOUTS, x_norm_pos + dx_pos + ( delta_char * x_ch_size ), $
          y_pos[ i ] - ( y_ch_size/2.0 ), $
          legend_text[ i ], $
          COLOR    = color[ i ], $
          CHARSIZE = charsize, $
          /NORM, $
          _EXTRA = extra
END

( psym[ i ] GT 0 ): BEGIN

  ; -- Just a symbol
  PLOTS, x_norm_pos, $
         y_pos[i], $
         PSYM    = ABS( psym[ i ] ), $
         COLOR   = color[ i ], $
         SYMSIZE = symsize[ i ], $
         /NORM, $
         _EXTRA = extra

  XYOUTS, x_norm_pos + ( delta_char * x_ch_size ), $
          y_pos[ i ] - ( y_ch_size/2.0 ), $
          legend_text[ i ], $
          COLOR    = color[ i ], $
          CHARSIZE = charsize, $
          /NORM, $
          _EXTRA = extra
```

```
      END

    ENDCASE

  ENDFOR


   ;------------------------------------------------------- -------------------
  ;                          -- Done --
   ;------------------------------------------------------- -------------------

  CATCH, /CANCEL
  RETURN

END


 ;------------------------------------------------------- -------------------
;                       -- MODIFICATION HISTORY --
 ;------------------------------------------------------- -------------------
;
; $Id:$
;
; $Date:$
;
; $Revision:$
;
; $State:$
;
; $Log:$
;
;
;
```