Subject: Re: Object Graphics Printing
Posted by btupper on Tue, 29 Oct 2002 02:14:56 GMT
View Forum Message <> Reply to Message

On Fri, 25 Oct 2002 07:57:40 -0600, David Fanning <david@dfanning.com>
wrote:

> Ben Tupper (btupper@bigelow.org) writes:
>
>> Now, if I could just figure out how David's neat little fsc_plotwindow
>> works!
>
> Yeah, me too. :-(
>
It really is a whiz-banger. Soo many details.

> Have you had a look at the FSC_SURFACE_PRINT module
> in FSC_SURFACE? I want the printed version of the
> window to have the same aspect ratio (I.e., look like)
> the version on the display, so I muck around with the
> printer aspect ratio, then set the view coordinates.
>

Good idea. I have modified your handy ASPECT program slightly
(appended below) so that the arguments are the VIEW and the
DESTINATION objects. Margins can be specified in each direction as a
two element vector (perhaps that should be 4 elements... left, bottom,
right, top?)

The tricky part with this is dealing with the default VIEW dimensions.
In this case, the programmer has allowed the VIEW dimensions to fit
the destination device (usually a window). A call to the VIEW's
getproperty method returns [0,0] for the dimensions of the view.
Oops, what kind of aspect is that? There's nothing to be done in
that case, except match the VIEW's dimensions to the DESTINATION (in
my case the printer - which doesn't have the same aspect ratio as the
VIEW.)

This makes it tough to come up with a general solution to using just a
VIEW to properly scale to the printer, buffer, clipboard,... unless
there's another way around it that is right under my nose!?!

Thanks for the tip,

Ben

```
;+
; NAME:
```

```
; FITASPECT
;
; PURPOSE:
; This function returns the new dimensions
;   required to fit a IDLgrView into an
;    IDLgrDestination device. The aspect ratio
;   of the orginal view is preserved if the VIEW
; are explicitly set.
;
; CATEGORY:
; Object Graphics
;
; CALLING SEQUENCE:
; returned = FITASPECT(view, $
; dest, [margin = margin], [location = location])
;
; ARGUMENTS:
; VIEW Set this to a valid IDLgrVIEW or an
;     object that is superclassed by IDLgrVIEW.
;     If the dimensions of the VIEW are not
;   explicitly set, these are returned by
;   IDL as [0.0, 0.0] which is the flag indicating
;     'fit to destination'.  In this case
;   the VIEW ASPECT will be figured to
;   fit the destination (accounting for margins.)
; DEST Set this to a valid IDLGRSRCDESTINATION
;  object, such as a Printer, Clipboard, Window, etc.
;
; KEYWORDS:
; MARGIN  Set this equal to two element
;     array of the x and y margins
;     in normalized units.  If set to a scalar
;     this margin is used in both the x and y directions.
;   The default is [0.0, 0.0].
; LOCATION  Set this equal to named
;     varibale to retrieve the appropriate
;     location for the view with in the destination.
;    This is a two element array of [xloc, yloc]
; UNITS Set this equal to a named variable
;     to retrieve the units of the
; destination device.
;
; EXAMPLE (kind of...)
; Suppose you had an object graphics view set
;   just the way you like it in and IDLgrWINDOW...
;   how do you place it on the printer and have it come
; with the same apsect ration and fitting on the page?
;
```

```
;  ;preserve you orginal settings
;  view->GetProperty, Dim = oldDim, Loc = oldLoc, units = oldUnits
;   ;get the new dimensions and location
;  newdim = fit_aspect(view, printer, margin = [0.1, 0.1], loc =
NewLoc, Units = newUnits)
;   ;update the view
; view->Setproperty, loc = newloc, dim = newdim, units = newunits
;   ;print
; printer->Draw, view
; printer, newdocument
;  ;restore the views orginal settings
;  view->SetProperty, Dim = oldDim, Loc = oldLoc, units = oldUnits
;
;
; MODIFICATION HISTORY:
;  Adapted from David Fanning's FSC_SURFACE and
; ASPECT programs.     28 OCT 2002, BT
;-


FUNCTION  Fit_Aspect, view, dest, $
 margin = margin, Location = Location, $
 Units = Units

If n_elements(View) EQ 0 Then $
 Message, 'View is required'

If Obj_ISA(View, 'IDLGRVIEW') NE 1 Then $
 Message, 'View must be IDLGRVIEW or '+ $
   'superclassed by IDLGRVIEW'

If n_elements(Dest) EQ 0 Then $
 Message, 'Destination is required'

If Obj_ISA(Dest,  'IDLGRSRCDEST') NE 1 Then $
 Message, 'View must be a destination class object ' + $
  'or superclassed by a destination class object'

 ;get requested margin in normalized units
Case n_elements(Margin) of
 0: Marg = [0.0, 0.0]
 1: Marg = [Margin, Margin]
 Else: Marg = Margin[0:1]
EndCase

 ;force the destination into device units
Dest->GetProperty, units = Units
Dest->SetProperty, units = 0
```

```
 ;get the destination properties
Dest->GetProperty, Dimension = dDim
dAspect = Float(dDim[1])/dDim[0]
Dest->SetProperty, units = Units

 ;now get the units again (in the original setting)
Dest->GetProperty, Dimension = dDim


 ;get the view properties
View->GetProperty, Dimension = vDim
If TOTAL(vDim) EQ 0 Then Begin
 Message, 'Dimensions of VIEW not explicitly set... '+ $
  'setting aspect to destination', /Info
 vAspect = dAspect
EndIf Else vAspect = Float(vDim[1])/vDim[0]

 ;compare the two
Case 1 Of

vAspect LE dAspect:  Begin

  ;taller than wide
  x0 = Marg[0]
  y0 = 0.5 - (0.5 - marg[1] ) * (vAspect/dAspect)
  x1 = 1.0 - marg[0]
  y1 = 0.5 + (0.5 -  marg[1] ) * (vAspect/dAspect)

  NewDim = [x1-x0, y1-y0] * dDim

 End

vAspect GT  dAspect : Begin

  ;wider than tall

 x0 = 0.5 - (0.5 - marg[0]) * (dAspect/vAspect)
 y0 = marg[1]
 x1 = 0.5 + (0.5 - marg[0]) * (dAspect/vAspect)
 y1 = 1.0 - marg[1]

 NewDim = [x1 - x0, y1 - y0] * dDim

 End

Else: Begin
```

```
 newDim = min(Region)
 NewDim = [NewDim, NewDim]

 End

EndCase

Location = (dDim - NewDim)/2.0

Return, NewDim
END ;
```