Subject: Re: Different Platforms
Posted by mgs on Wed, 19 Nov 1997 08:00:00 GMT
View Forum Message <> Reply to Message

In article <MPG.edc959fe74be3f79896ac@news.frii.com>, davidf@dfanning.com (David Fanning) wrote:

> Neil Winrow (ncw@dl.ac.uk) writes:
>
>> I have written a number of widget programs, which are visually correct
>> on my PC, however when they are run on the silicon graphics machines the
>> layout starts to go terribly wrong. The character sizes are wrong, and
>> the labelling carried out using the 'XYOUTS' call is all wrong. The
>> whole window sizing falls down. The programs are going to be used on
>> PC's, silicon graphics, and MAC's. Could anyone offer me a few pointers
>> on how to correct these problems to run the programs on the different
>> platforms.
>
> There is nothing quite like trying to get IDL widget programs
> to run on PCs, SGIs and Macs to get you to question your
> sanity. And, of course, as soon as you have it figured out,
> someone buys a Sun and then you have to deal with *those*
> tinsy-tiny 12-point fonts. You soon realize that "font size"
> must be one of those words like "true love" that is open to
> all kinds of interpretation.
...

> Let us know what you come up with. I'll add it to the list.
> It will make interesting reading 100 years from now. :-)

I put together a couple functions to help out with this. One returns a structure based on !Version.OS_Family which contains offsets for widget parameters such as scroll bars, frames, and button types, with adjustments based on the window manager. The other creates a set of fontnames which can be called generically cross-platform. Sizing of widgets also relies on calls to Widget_info with the /Geometry keyword set. It's ugly, but useful.

```
   mOsInfo = {$
      sDiskChar: '', $       ;   disk separator character
      sDirChar: '', $        ;   directory separator character
      sWinName: '', $        ;   Window manager name
      mBuffer: {$            ;   buffer offsets
         scroll: 0, $        ;   scroll bar
         frame: 0, $         ;   frame boundary
         exclusive: 0, $     ;   radio button (exclusive)
         nonexclusive: 0, $  ;   checkbox button (nonexclusive)
         button: 0}, $       ;   button (regular)
```

```
   ;   followed by settings for each platform - UNIX, Mac, Windows, VMS

FontGen returns a structure of fonts for proportional and monotype sizes
of 10, 12, 18, and 24 at medium and bold weights. The default fonts are
times and courier but can be whatever you specify. The referenced font
names are based on the platform you are running IDL on: Mac/PC mono10b
would be "courier*bold*10", UNIX mono10b would be
"*-courier-bold-r-normal--10-*"

   mFont = FontGen()

   ;   declare some generic font names for common usage
   labelFont = mFont.prop12b ; proportional (times), 12 point, bold
   buttonFont = mFont.prop12m ; proportional, 12 point, medium
   fieldFont = mFont.mono10m ; monotype (courier), 10 point, medium

   ;    the mMisc structure contains mFont as well as mOsInfo
   wLabel = Widget_Label(wBase, Value='Output on Close: ', $
      Font=mMisc.labelFont)

   ;   get dimensions of base
   mGeoBase = Widget_Info(wBase, /Geometry)

   ;   get dimensions of wLabel
   mGeoLabel = Widget_Info(wLabel, /Geometry)

   asWriteText = ['Trend', 'Report', 'Residuals']

   ;   display the write buttons
   ;   CW_BGroup2 is a modification of CW_BGroup that allows
   ;   additional font, and sizing info to be set.
   wBGWrite = CW_BGroup2(wBaseOutput, asWriteText, /Row, /Frame, $
      /Align_Left, ButtonSize=((mGeoBase.XSize - mGeoLabel.XSize) / $
      N_Elements(asWriteText) - mMisc.mOsInfo.mBuffer.nonexclusive - $
      mMisc.mOsInfo.mBuffer.frame /  N_Elements(asWriteText)), $
      /NonExclusive, ButtonFont=mMisc.buttonFont)

---------------------- wBase ----------------------
          ----------------------------------------
Output OC: | [] Trend   []   Report [] Residuals  |
          ----------------------------------------
 wLabel                    wBGWrite
-------------------------------------------------

assuming:
wBase = 400 pixels (from Widget_Info(wBase, /Geometry))
wLabel = 130 pixels (from Widget_Info(wBase, /Geometry))
nonexclusive = 16 pixels (width of a non-exclusive button)
```

frame = 6 pixels (width of a frame - both sides)
N_Elements(asWriteText) = 3

buttonsize = ((wbase - wlabel) / 3) - nonexclusive - frame / 3
buttonsize = ((400 - 130) / 3) - 16 - 6 / 3 = 72 pixels
The framed group will fit in the base next to the label as shown above.

Since I have settings for the values of nonexclusive and frame in my mOsInfo structure and the Widget_info(widget_id, /Geometry) call returns the current size of the specified widgets, I have a platform-independent sizing system.

--
Mike Schienle Interactive Visuals
mgs@sd.cybernex.net http://ww2.sd.cybernex.net/~mgs/