Subject: Widget_base woes
Posted by Richard French on Thu, 24 Nov 2005 06:50:08 GMT
View Forum Message <> Reply to Message

I'm going crazy trying to lay out a bunch of widgets. The problem is that I
am having trouble getting the widget_base() commands done in such a way that
the widgets appear where I want. I think part of my problem is in trying to
sort out what the row and column keywords really mean.

Here is roughly what I want:

2 512x512 draw widgets across the top (the second with scroll bars),
followed by a third tall, skinny draw widget with scroll bars that will be
the full extent of the entire widget frame.

Below the left draw widgets, I want four cw_fsliders.

Below that, I want two columns of four widget sliders.

To the right of that, I want a set of buttons.

I get get pretty much everything working EXCEPT that I can't seem to figure
out how to have the long, skinny vertical draw widget to the right of
everything else.

I'm not asking someone to do the design for me, but I would appreciate some
hints on how to set up the successive hierarchy of widget_base calls so that
I can deterministically figure out where the widgets will go, instead of
blindly setting column=2 or row=1 and hoping for the best.

Thanks for any insight you can provide into this. I can't use the GUI
builder since it exists only on Windows now.

Dick French

Subject: Re: Widget_base woes
Posted by peter.albert@gmx.de on Thu, 24 Nov 2005 08:03:26 GMT
View Forum Message <> Reply to Message

Hi Dick,

did you ever create HTML pages using tables? That sure helps, as it is
the same principle of nesting. The problem with widgets is that the ROW
and COLUMN keywords just work weired. Setting ROW=n will create n rows
in the end, however, subseqent elements are put in subsequent columns.
Just an example:

```
base=widget_base(row=3)
a=widget_button(base, value="a")
b=widget_button(base, value="b")
c=widget_button(base, value="c")
d=widget_button(base, value="d")
widget_control, base, /real
```

will create a widget which looks loke this:

```
a   b
c
d
```

So we have three rows, as we defined, but not that the buttons were filled in row after row, that woul have been too easy ...

As for your special problem, the key is the subsequent useage of base widgets. I have written an example just using widget buttons just to illustrate the main points:

```
; This is the main widget, it has two columns, one for the
; skinny draw widget on the right, one for all the rest.

  base = widget_base(col=2)

; All the left parts go into their own base widget,
; which will have 3 rows:
; The draw widgets, the cw_sliders, and another base widget with
; more sliders and buttons:

  left = widget_base(base, row=3)

; Here comes the skinny draw widget, based in the root widget:
  skinny = widget_button( $
                        base, $
                        value="skinny draw widget", $
                        xsize=100, ysize=300 $
                      )


; Now for the draw widgets, based in the "left" widget. Just to be sure
; we use another base widget where we specify that all
; draw widgets  go into one row:

  draw = widget_base(left, row=1)
  d1 = widget_button(draw, value="first draw widget")
  d2 = widget_button(draw, value="second draw widget")
```

```
; Now for the sliders: same approach:
  sliders = widget_base(left, row=1)
  s1 = widget_button(sliders, value = '1st slider')
  s2 = widget_button(sliders, value = '2nd slider')
  s3 = widget_button(sliders, value = '3rd slider')
  s4 = widget_button(sliders, value = '4th slider')

; Now the sliders and buttons are more complicated, we
; need another base widgets holding in turn the left part
; (the columns of sliders) and the right part (the set of buttons)
; This base widget has only one row, as it will only contain
; two more base widgets:

  sliders_and_buttons = widget_base(left, row=1)

; This is the base widget for the sliders. You want two columns with
; four rows here, so we could set either row = 4 or columns = 2, but
; we use col = 2 as we want subsequent sliders appear beneath
; each other:

  sliders = widget_base(sliders_and_buttons, col = 2)
  s1 = widget_button(sliders, value = '1st slider')
  s2 = widget_button(sliders, value = '2nd slider')
  s3 = widget_button(sliders, value = '3rd slider')
  s4 = widget_button(sliders, value = '4th slider')
  s5 = widget_button(sliders, value = '5th slider')
  s6 = widget_button(sliders, value = '6th slider')
  s7 = widget_button(sliders, value = '7th slider')
  s8 = widget_button(sliders, value = '8th slider')

: Now the set of buttons is to appear in one column:

  buttons = widget_base(sliders_and_buttons, col = 1)
  b1 = widget_button(buttons, value = '1st button')
  b2 = widget_button(buttons, value = '2nd button')
  b3 = widget_button(buttons, value = '3rd button')


; That's it :-)
  widget_control, base, /real
```

Cheers,

Peter

Subject: Re: Widget_base woes
Posted by David Fanning on Thu, 24 Nov 2005 14:46:29 GMT
View Forum Message <> Reply to Message

Richard G. French writes:

> I'm going crazy trying to lay out a bunch of widgets. The problem is that I
> am having trouble getting the widget_base() commands done in such a way that
> the widgets appear where I want. I think part of my problem is in trying to
> sort out what the row and column keywords really mean.

I agree with Peter that laying out widgets is like creating
an HTML table, but I can probably count on one hand the times
I've used a COLUMN or ROW keyword set to something other than 1.
I just use lots and lots of COLUMN=1 and ROW=1 bases to get
the layout I want.

Bases are not suppose to take any space, but I notice they
do in Windows XP, so I usually set XPAD=0, YPAD=0 in these
organizing bases, too.

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Widget_base woes
Posted by Richard French on Thu, 24 Nov 2005 15:02:18 GMT
View Forum Message <> Reply to Message

"Peter Albert" <peter.albert@gmx.de> wrote:

> did you ever create HTML pages using tables? That sure helps, as it is
> the same principle of nesting. The problem with widgets is that the ROW
> and COLUMN keywords just work weired. Setting ROW=n will create n rows
> in the end, however, subseqent elements are put in subsequent columns.

Dear Peter:
Many thanks for your very lucid explanation! I am going to digest it along
with my Thanksgiving Day turkey. I especially appreciate the very first part
of your answer, in which you show how (and why) the strange example fails in
a simple case. I use this all the time in my own classes: "You might think
that this would be a great place to use Fourier series, but let's see why it
isn't." Students get a lot more out of seeing why a plausible but wrong

solution is wrong, than by having the prof say, 'We'll let u =tanh^-1 (y^2),
and then the differential equation looks just like a forced harmonic
oscillator' - Uh, how did you do that????

I'm very grateful to you.

Dick French

Subject: Re: Widget_base woes
Posted by David Fanning on Thu, 24 Nov 2005 15:28:36 GMT
View Forum Message <> Reply to Message

Peter Albert writes:

> The problem with widgets is that the ROW
> and COLUMN keywords just work weired. Setting ROW=n will create n rows
> in the end, however, subseqent elements are put in subsequent columns.
> Just an example:
>
> base=widget_base(row=3)
> a=widget_button(base, value="a")
> b=widget_button(base, value="b")
> c=widget_button(base, value="c")
> d=widget_button(base, value="d")
> widget_control, base, /real
>
> will create a widget which looks loke this:
>
> a b
> c
> d
>
> So we have three rows, as we defined, but not that the buttons were
> filled in row after row, that woul have been too easy ...

I agree that setting COLUMN and ROW keywords to something
other than 1 results in screwy results most of the time,
but I think the algorithm it uses for layout is a reasonable
algorithm. It seems to be something like this for a ROW base:

1. Calculate the X size of the widgets that are going to go
   into this base.
2. Maintaining their creation order (as I do everywhere else),
   put them in X rows in such a way that I get approximately
   X equal-length rows.

The problems comes about when you put things in the base that
do not have identical sizes. Since widgets are discrete things
and cannot be divided, the layout can be surprising. And it will
also be affected (usually in a negative way) by other things around
it.

In addition to sticking to COLUMN=1 and ROW=1 keywords, I sometimes
use the GRID keyword, too. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
