
Subject: Re: Resizable IDL List widget?

Posted by [David Fanning](#) on Fri, 06 Apr 2007 20:01:46 GMT

[View Forum Message](#) <> [Reply to Message](#)

marty.hu@gmail.com writes:

- > I'm working on an IDL Widget Program that reads a number of images and
- > outputs some of their features into a new window containing a
- > scrollable IDL list widget. Is it possible to drag-resize the list
- > widget to be larger/smaller so the user will have more flexibility?

I assume you mean "smoothly, across all architectures." :-)

It doesn't really matter what you mean, I don't know the answer. But if someone was forcing me to bet big money, I'd probably bet NO, it isn't possible. It just doesn't seem to me like the kind of thing IDL widgets can do.

You probably could figure out a way to hack the functionality, though, especially if by "new window" you mean a list widget in its own top-level base widget:

http://www.dfanning.com/widget_tips/dynamic_menus.html

Cheers,

David

--

David Fanning, Ph.D.

Fanning Software Consulting, Inc.

Coyote's Guide to IDL Programming: <http://www.dfanning.com/>

Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Resizable IDL List widget?

Posted by [Michael Galloy](#) on Fri, 06 Apr 2007 22:18:02 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Apr 6, 12:27 pm, marty...@gmail.com wrote:

- > Hi All,
- >
- > I'm working on an IDL Widget Program that reads a number of images and
- > outputs some of their features into a new window containing a
- > scrollable IDL list widget. Is it possible to drag-resize the list
- > widget to be larger/smaller so the user will have more flexibility?
- >
- > Thanks,
- > Marty

There is the `cw_panes` compound widget. It is not documented in the online help, but the header file for it is fairly useful. It is in `$IDL_DIR/lib/itools/ui_widgets/`.

To see it in action, start an iTool:

```
iplot, findgen(11)
```

Double click on the plot. That should bring up the Visualization Browser. Click on the arrow in the upper left corner. Now you can resize between the tree on the left and the property sheet on the right by dragging the bar in the middle. This is written in IDL.

It does seem to have iTools dependencies, but in `CW_PANES_EVENT` only. I tried once to use this in non-iTools code and got the basics to work fairly well, but the final product wasn't quite good enough.

So...your results may vary.

Mike

--

www.michaelgalloy.com

Subject: Re: Resizable IDL List widget?

Posted by [JD Smith](#) on Sat, 07 Apr 2007 00:56:28 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Fri, 06 Apr 2007 13:01:46 -0700, David Fanning wrote:

> marty.hu@gmail.com writes:

>

>> I'm working on an IDL Widget Program that reads a number of images and

>> outputs some of their features into a new window containing a scrollable

>> IDL list widget. Is it possible to drag-resize the list widget to be

>> larger/smaller so the user will have more flexibility?

>

> I assume you mean "smoothly, across all architectures." :-)

>

> It doesn't really matter what you mean, I don't know the answer. But if

> someone was forcing me to bet big money, I'd probably bet NO, it isn't

> possible. It just doesn't seem to me like the kind of thing IDL widgets

> can do.

Well, I don't know if it's really cross-platform (which, these days, simply means "does it support Windows?", since all the other widget toolkits on all other architectures are exactly the same), but I have

a method which works well for me.

It consists of computing the difference in pixels in screen size between a list widget and its containing TLB just after it has been realized, storing that difference, and then applying it to resize-generated TLB_SIZE_EVENTS sizes as they come in. It works quite generally, even if there are other widgets inside the same base above, adjacent, below, etc. It effectively "subtracts out" all the random and variable frame size, widget spacing, and other widget display difference which vary by OS, window manager, etc. As a result, it should, at least in principle, be cross-platform (untested). Compile and run RESIZE_LIST:

```
;;-----  
;; Resizable List Widget Test, JDS, 04/2007  
pro resize_list_event, ev  
  widget_control, ev.id, GET_UVALUE=uval  
  if size(uval, /TYPE) eq 7 then begin  
    if uval eq 'quit' then begin  
      widget_control, ev.top, /DESTROY  
      return  
    endif  
  endif else begin  
    if tag_names(ev, /STRUCTURE_NAME) eq 'WIDGET_BASE' then begin  
      widget_control, uval.list, SCR_XSIZE=ev.X+uval.diff[0], $  
        SCR_YSIZE=ev.Y+uval.diff[1]  
    endif  
  endelse  
end  
  
pro resize_list  
  b=widget_base(/COLUMN, SPACE=0)  
  t=widget_label(b, VALUE='A Resizable list widget by JDS')  
  r=widget_base(b, /ROW, SPACE=2, /BASE_ALIGN_CENTER)  
  d=widget_draw(r, UVALUE='draw', XSIZE=100, YSIZE=100)  
  text=(byte('a'))[0]+byte(randomu(sd, 50, 100)*26)  
  text[long(randomu(sd, 5*100)*50*100)]=32b  
  text=string(text)  
  l=widget_list(r, XSIZE=40, YSIZE=10, UVALUE='list', VALUE=text)  
  q=widget_button(b, VALUE='Quit', UVALUE='quit')  
  widget_control, b, /REALIZE, /TLB_SIZE_EVENTS  
  
  widget_control, d, GET_VALUE=dw  
  wset, dw  
  tvscl, dist(100)  
  xyouts, 50, 10, 'Random Graphic', /DEVICE, ALIGNMENT=0.5  
  
  geom=widget_info(l, /GEOMETRY)
```

```

bgeom=widget_info(b,/GEOMETRY)
list_size_diff=[geom.SCR_XSIZE-bgeom.SCR_XSIZE,geom.SCR_YSIZ E-bgeom.SCR_YSIZE]
state={diff:list_size_diff,list:l}
widget_control, b,SET_UVALUE=state
XManager,'resize_list',b
end
;;-----

```

Now, what if you have more than one list (or draw widget, etc.) that you want to resize, all within a single TLB? You could easily extend this formalism as follows. After realizing the widget, store the difference between the TLB screen geometry, and the *sum* of sizes of adjacent resizeable widgets within it (e.g. only if horizontally adjacent should the xsizes be summed, etc.). When a resize event comes in, compute the new total size in each direction, and then distribute the changed size in some way among all the desired resizeable widgets inside it (wrapping in UPDATE=0, UPDATE=1 to prevent screen flicker).

You could of course use any sort of custom algorithm to determine which widget gets which fraction of the new size, but one simple method is just to preserve the original size ratio(s) among the relevant widget dimensions. A simple addition to this method would be to resize one up to some limit according to the original size ratio, and then fix it thereafter. Obviously the sky is the limit with how you distribute the wealth (of new screen real estate). Here's an example of the "fixed ratio, up to some maximum list width (here 400 pixels)" method, where I'm matching the ysize of the list and draw widgets, and for fun re-generating the image as well. When it says "List Pegged" that means the list is as wide as it will get.

Good luck,

JD

```

;;-----
;; Resizeable List and Draw Widget Test, JDS, 04/2007
pro resize_list_draw_event,ev
  widget_control, ev.id,GET_UVALUE=uval
  if size(uval,/TYPE) eq 7 then begin
    if uval eq 'quit' then begin
      widget_control, ev.top,/DESTROY
      return
    endif
  endif else begin
    if tag_names(ev,/STRUCTURE_NAME) eq 'WIDGET_BASE' then begin
      widget_control, ev.top,UPDATE=0
      new_y=ev.Y+uval.diff[1]
    endif
  endif
end

```

```

new_x=ev.X+uval.diff[0]
new_x_l=long(new_x*uval.frac)
pegged=new_x_l gt 400
new_x_l<=400      ;maximum list width
new_x_d=new_x-new_x_l
widget_control, uval.list,SCR_XSIZE=new_x_l, SCR_YSIZE=new_y
widget_control, uval.draw,SCR_XSIZE=new_x_d, SCR_YSIZE=new_y, $
    GET_VALUE=dw
wset,dw
widget_control, ev.top,/UPDATE
tvsc1,dist(new_x_d,new_y)
xyouts,new_x_d/2,10, /DEVICE, ALIGNMENT=0.5,$
    'Random Graphic'+(pegged?" (List pegged)": "")
endif
endelse
end

pro resize_list_draw
b=widget_base(/COLUMN,SPACE=0)
t=widget_label(b,VALUE='A Resizeable list+draw widget by JDS')
r=widget_base(b,/ROW,SPACE=2,/BASE_ALIGN_CENTER)
d=widget_draw(r,UVALUE='draw',XSIZE=100,YSIZE=100)
text=(byte('a'))[0]+byte(randomu(sd,50,100)*26)
text[long(randomu(sd,5*100)*50*100)]=32b
text=string(text)
l=widget_list(r,XSIZE=40,YSIZE=10,UVALUE='list', VALUE=text)
q=widget_button(b,VALUE='Quit',UVALUE='quit')
widget_control, b,/REALIZE,/TLB_SIZE_EVENTS

lgeom=widget_info(l,/GEOMETRY)
bgeom=widget_info(b,/GEOMETRY)
widget_control, d, SCR_YSIZE=lgeom.SCR_YSIZE
dgeom=widget_info(d,/GEOMETRY)

;; They are adjacent in X, but not in Y
list_size_diff=[lgeom.SCR_XSIZE+dgeom.SCR_XSIZE-bgeom.SCR_XSIZE, $
    lgeom.SCR_YSIZE-bgeom.SCR_YSIZE]
state={diff:list_size_diff,list:l,draw:d, $
    frac:float(lgeom.SCR_XSIZE)/(lgeom.SCR_XSIZE+dgeom.SCR_XSIZE )}

widget_control, d, GET_VALUE=dw
wset,dw
tvsc1,dist(100,lgeom.SCR_YSIZE)
xyouts,50,10,'Random Graphic',/DEVICE,ALIGNMENT=0.5

widget_control, b,SET_UVALUE=state
XManager,'resize_list_draw',b,/NO_BLOCK
end

```

;;-----

Subject: Re: Resizable IDL List widget?

Posted by [David Fanning](#) on Sun, 08 Apr 2007 02:05:10 GMT

[View Forum Message](#) <> [Reply to Message](#)

JD Smith writes:

- > Well, I don't know if it's really cross-platform (which, these days,
- > simply means "does it support Windows?", since all the other widget
- > toolkits on all other architectures are exactly the same), but I have
- > a method which works well for me.

Yes, more or less what I had in mind. On my Windows machines, though, I have to add "fudge factors" of 26 in the Y direction and 6 in the X direction to make the window "stick" to the size I set it to. The fudge factors correspond, roughly, to the vertical size of the title bar, and the extra horizontal padding size of base widgets, although when I try to take all this into account programmatically, I STILL end up with fudge factors I can't explain.

But, sometimes, "close enough" is close enough, and I could probably live with this solution if I could just train myself to overshoot by a little bit. :-)

Cheers,

David

--

David Fanning, Ph.D.

Fanning Software Consulting, Inc.

Coyote's Guide to IDL Programming: <http://www.dfanning.com/>

Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Resizable IDL List widget?

Posted by [marty.hu](#) on Tue, 10 Apr 2007 00:42:43 GMT

[View Forum Message](#) <> [Reply to Message](#)

Thanks all for the help and suggestions. I will definitely take a look at the `resize_list` procedure.

Marty

Subject: Re: Resizable IDL List widget?

Posted by [JD Smith](#) on Thu, 12 Apr 2007 19:08:41 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Sat, 07 Apr 2007 19:05:10 -0700, David Fanning wrote:

> JD Smith writes:

>

>> Well, I don't know if it's really cross-platform (which, these days,
>> simply means "does it support Windows?", since all the other widget
>> toolkits on all other architectures are exactly the same), but I have
>> a method which works well for me.

>

> Yes, more or less what I had in mind. On my Windows machines,
> though, I have to add "fudge factors" of 26 in the Y direction
> and 6 in the X direction to make the window "stick" to the
> size I set it to. The fudge factors correspond, roughly,
> to the vertical size of the title bar, and the extra
> horizontal padding size of base widgets, although when I
> try to take all this into account programmatically, I
> STILL end up with fudge factors I can't explain.

>

> But, sometimes, "close enough" is close enough, and I could
> probably live with this solution if I could just train myself
> to overshoot by a little bit. :-)

So, you tried my method on Windows and needed to add offsets? Or is this just a general statement? I'd be interested to know if, indeed, it works as expected under Windows.

JD
