Subject: widget_problem
Posted by d.poreh on Wed, 16 Jul 2008 17:11:53 GMT
View Forum Message <> Reply to Message

Folks
I have written a widget that read two arrays. when I want to access to
that arrays in another *pro* in that widget it can't recognize them.
My widget is like this:
I have a *file* button to read this 2 arrays (read elevation + read
area) when I want to call this two array from another button say*
analyze *, an error arising that says cannot recognize the arrays. I
am binger in the widget programming may be this question is stupid but
I can't fix it.
Any help in advance greatly will be appreciated
Cheers

Subject: Re: widget_problem
Posted by Paul Van Delst[1] on Thu, 17 Jul 2008 18:32:37 GMT
View Forum Message <> Reply to Message

Justus Skorps wrote:
>> Justus
>> stil can not fix it. i have Liam E.Gumley's book but .....
>> Cheers
>> Dave
>
> after u load your arrays (lets call them A) store them with
>
> widget_control, event.top, set_uvalue=A, /nocopy
>
> You have to change 'event.top' that it fits your program...
> In your second button you can now load the arrays with
>
> widget_control, event.top, get_uvalue=A, /nocopy

but don't forget to put them back into your uvalue when you're done!
>
> It is useful to
>
> -store the data in the main widget
> -use a structure to store every data you want

I also tend to use procedures to get the Info state:

; Routine to get the Info state
PRO GetState, ID, Info, No_Copy = No_Copy
  ; -- Get pointer

```
  WIDGET_CONTROL, ID, GET_UVALUE = InfoPtr
  IF ( PTR_VALID( InfoPtr ) EQ 0 ) THEN $
    MESSAGE, 'State Information pointer is invalid'

  ; -- Get state information structure
  IF ( N_ELEMENTS( *InfoPtr ) EQ 0 ) THEN $
    MESSAGE, 'State information structure is undefined'
  IF ( KEYWORD_SET( No_Copy ) ) THEN BEGIN
    Info = TEMPORARY( *InfoPtr )
  ENDIF ELSE BEGIN
    Info = *InfoPtr
  ENDELSE
  IF ( Info.Debug EQ 1 ) THEN PRINT, 'GetState'
END
```

and to set the info state

```
; Routine to set the Info state
PRO SetState, ID, Info, No_Copy = No_Copy
  ; -- Get pointer
  WIDGET_CONTROL, ID, GET_UVALUE = InfoPtr
  IF ( PTR_VALID( InfoPtr ) EQ 0 ) THEN $
  MESSAGE, 'State information pointer is invalid'

  ; -- Set state information structure
  IF ( N_ELEMENTS( Info ) EQ 0 ) THEN $
    MESSAGE, 'State information structure is undefined'
  IF ( KEYWORD_SET( No_Copy ) ) THEN BEGIN
    *InfoPtr = TEMPORARY( Info )
  ENDIF ELSE BEGIN
    *InfoPtr = Info
  ENDELSE
  IF ( (*InfoPtr).Debug EQ 1 ) THEN PRINT, 'SetState'
END
```

My widget event handlers then do something like:

```
FUNCTION ComponentTest_LogLin_Event, Event
  ; -- Get main info state
  GetState, Event.Top, Info

  ; -- Print debug statement if required
  IF ( Info.Debug EQ 1 ) THEN PRINT, 'ComponentTest_LogLin_Event'
  ; -- Set the selected variable number index
  Info.LogLin_Index = Event.Value
```

```
  ; -- Save info state
  SetState, Event.Top, Info

  ; -- Display the result
  ComponentTest_Display, Event.Top
  RETURN, 0
END
```

Note how I call GetState and then SetState. Because I tend not to use /no_copy, it's really only an issue when I update a component of the info state (like in my example above). But, if you *do* use /no_copy, then I think you have to call SetState again to replace the info pointer.

cheers,

paulv

Subject: Re: widget_problem
Posted by d.poreh on Tue, 22 Jul 2008 15:52:35 GMT
View Forum Message <> Reply to Message

On Jul 17, 8:32 pm, Paul van Delst <Paul.vanDe...@noaa.gov> wrote:
> Justus Skorps wrote:
>>> Justus
>>> stil can not fix it. i have Liam E.Gumley's book but .....
>>> Cheers
>>> Dave
>
>> after u load your arrays (lets call them A) store them with
>
>> widget_control, event.top, set_uvalue=A, /nocopy
>
>> You have to change 'event.top' that it fits your program...
>> In your second button you can now load the arrays with
>
>> widget_control, event.top, get_uvalue=A, /nocopy
>
> but don't forget to put them back into your uvalue when you're done!
>
>
>
>> It is useful to
>
>> -store the data in the main widget
>> -use a structure to store every data you want

```
>
>  I also tend to use procedures to get the Info state:
>
>  ; Routine to get the Info state
>  PRO GetState, ID, Info, No_Copy = No_Copy
>    ; -- Get pointer
>    WIDGET_CONTROL, ID, GET_UVALUE = InfoPtr
>    IF ( PTR_VALID( InfoPtr ) EQ 0 ) THEN $
>      MESSAGE, 'State Information pointer is invalid'
>
>    ; -- Get state information structure
>    IF ( N_ELEMENTS( *InfoPtr ) EQ 0 ) THEN $
>      MESSAGE, 'State information structure is undefined'
>    IF ( KEYWORD_SET( No_Copy ) ) THEN BEGIN
>      Info = TEMPORARY( *InfoPtr )
>    ENDIF ELSE BEGIN
>      Info = *InfoPtr
>    ENDELSE
>    IF ( Info.Debug EQ 1 ) THEN PRINT, 'GetState'
>  END
>
>  and to set the info state
>
>  ; Routine to set the Info state
>  PRO SetState, ID, Info, No_Copy = No_Copy
>    ; -- Get pointer
>    WIDGET_CONTROL, ID, GET_UVALUE = InfoPtr
>    IF ( PTR_VALID( InfoPtr ) EQ 0 ) THEN $
>    MESSAGE, 'State information pointer is invalid'
>
>    ; -- Set state information structure
>    IF ( N_ELEMENTS( Info ) EQ 0 ) THEN $
>      MESSAGE, 'State information structure is undefined'
>    IF ( KEYWORD_SET( No_Copy ) ) THEN BEGIN
>      *InfoPtr = TEMPORARY( Info )
>    ENDIF ELSE BEGIN
>      *InfoPtr = Info
>    ENDELSE
>    IF ( (*InfoPtr).Debug EQ 1 ) THEN PRINT, 'SetState'
>  END
>
>  My widget event handlers then do something like:
>
>  FUNCTION ComponentTest_LogLin_Event, Event
>    ; -- Get main info state
>    GetState, Event.Top, Info
>
>    ; -- Print debug statement if required
```

> IF (Info.Debug EQ 1) THEN PRINT, 'ComponentTest_LogLin_Event'
> ; -- Set the selected variable number index
> Info.LogLin_Index = Event.Value
>
> ; -- Save info state
> SetState, Event.Top, Info
>
> ; -- Display the result
> ComponentTest_Display, Event.Top
> RETURN, 0
> END
>
> Note how I call GetState and then SetState. Because I tend not to use /no_copy, it's
> really only an issue when I update a component of the info state (like in my example
> above). But, if you *do* use /no_copy, then I think you have to call SetState again to
> replace the info pointer.
>
> cheers,
>
> paulv

Thanks Justus and Paulv
You help me very much. And it was very useful.
Cheers
Dave

Subject: Re: widget_problem
Posted by d.poreh on Wed, 23 Jul 2008 14:01:08 GMT
View Forum Message <> Reply to Message

On Jul 22, 5:52 pm, d.po...@gmail.com wrote:
> On Jul 17, 8:32 pm, Paul van Delst <Paul.vanDe...@noaa.gov> wrote:
>
>
>
>> Justus Skorps wrote:
>>>> Justus
>>>> stil can not fix it. i have Liam E.Gumley's book but .....
>>>> Cheers
>>>> Dave
>
>>> after u load your arrays (lets call them A) store them with
>
>>> widget_control, event.top, set_uvalue=A, /nocopy
>
>>> You have to change 'event.top' that it fits your program...
>>> In your second button you can now load the arrays with

>
>>> widget_control, event.top, get_uvalue=A, /nocopy
>
>> but don't forget to put them back into your uvalue when you're done!
>
>>> It is useful to
>
>>> -store the data in the main widget
>>> -use a structure to store every data you want
>
>> I also tend to use procedures to get the Info state:
>
>> ; Routine to get the Info state
>> PRO GetState, ID, Info, No_Copy = No_Copy
>>   ; -- Get pointer
>>   WIDGET_CONTROL, ID, GET_UVALUE = InfoPtr
>>   IF (PTR_VALID(InfoPtr) EQ 0) THEN $
>>   MESSAGE, 'State Information pointer is invalid'
>
>>   ; -- Get state information structure
>>   IF (N_ELEMENTS(*InfoPtr) EQ 0) THEN $
>>   MESSAGE, 'State information structure is undefined'
>>   IF (KEYWORD_SET(No_Copy)) THEN BEGIN
>>   Info = TEMPORARY(*InfoPtr)
>>   ENDIF ELSE BEGIN
>>   Info = *InfoPtr
>>   ENDELSE
>>   IF (Info.Debug EQ 1) THEN PRINT, 'GetState'
>> END
>
>> and to set the info state
>
>> ; Routine to set the Info state
>> PRO SetState, ID, Info, No_Copy = No_Copy
>>   ; -- Get pointer
>>   WIDGET_CONTROL, ID, GET_UVALUE = InfoPtr
>>   IF (PTR_VALID(InfoPtr) EQ 0) THEN $
>>   MESSAGE, 'State information pointer is invalid'
>
>>   ; -- Set state information structure
>>   IF (N_ELEMENTS(Info) EQ 0) THEN $
>>   MESSAGE, 'State information structure is undefined'
>>   IF (KEYWORD_SET(No_Copy)) THEN BEGIN
>>   *InfoPtr = TEMPORARY(Info)
>>   ENDIF ELSE BEGIN
>>   *InfoPtr = Info
>>   ENDELSE
>>   IF ((*InfoPtr).Debug EQ 1) THEN PRINT, 'SetState'

>> END
>
>> My widget event handlers then do something like:
>
>> FUNCTION ComponentTest_LogLin_Event, Event
>> ; -- Get main info state
>> GetState, Event.Top, Info
>
>> ; -- Print debug statement if required
>> IF (Info.Debug EQ 1) THEN PRINT, 'ComponentTest_LogLin_Event'
>> ; -- Set the selected variable number index
>> Info.LogLin_Index = Event.Value
>
>> ; -- Save info state
>> SetState, Event.Top, Info
>
>> ; -- Display the result
>> ComponentTest_Display, Event.Top
>> RETURN, 0
>> END
>
>> Note how I call GetState and then SetState. Because I tend not to use /no_copy, it's
>> really only an issue when I update a component of the info state (like in my example
>> above). But, if you *do* use /no_copy, then I think you have to call SetState again to
>> replace the info pointer.
>
>> cheers,
>
>> paulv
>
> Thanks Justus and Paulv
> You help me very much. And it was very useful.
> Cheers
> Dave

Hi Justus
I encounter a now problem:
Say I have set and get 2 arrays by this method:
widget_control, event.top, set_uvalue=A, /nocopy
…
widget_control, event.top, get_uvalue=A, /nocopy

and from another button:
widget_control, event.top, set_uvalue=B, /nocopy
…
widget_control, event.top, get_uvalue=B, /nocopy

Now I want to get both A&B in another button like this:

 widget_control, event.top, get_uvalue=A, /nocopy
widget_control, event.top, get_uvalue=B, /nocopy

But widget just accepts one array. How I can take this two arrays
simultaneously in another button?
Any help?
Cheers
Dave

Subject: Re: widget_problem
Posted by Paul Van Delst[1] on Wed, 23 Jul 2008 14:34:13 GMT
View Forum Message <> Reply to Message

d.poreh@gmail.com wrote:
> On Jul 22, 5:52 pm, d.po...@gmail.com wrote:
>> On Jul 17, 8:32 pm, Paul van Delst <Paul.vanDe...@noaa.gov> wrote:
>>
>>
>>
>>> Justus Skorps wrote:
>>>> > Justus
>>>> > stil can not fix it. i have Liam E.Gumley's book but .....
>>>> > Cheers
>>>> > Dave
>>>> after u load your arrays (lets call them A) store them with
>>>> widget_control, event.top, set_uvalue=A, /nocopy
>>>> You have to change 'event.top' that it fits your program...
>>>> In your second button you can now load the arrays with
>>>> widget_control, event.top, get_uvalue=A, /nocopy
>>> but don't forget to put them back into your uvalue when you're done!
>>>> It is useful to
>>>> -store the data in the main widget
>>>> -use a structure to store every data you want
>>> I also tend to use procedures to get the Info state:
>>> ; Routine to get the Info state
>>> PRO GetState, ID, Info, No_Copy = No_Copy
>>>   ; -- Get pointer
>>>   WIDGET_CONTROL, ID, GET_UVALUE = InfoPtr
>>>   IF (PTR_VALID(InfoPtr) EQ 0) THEN $
>>>   MESSAGE, 'State Information pointer is invalid'
>>>   ; -- Get state information structure
>>>   IF (N_ELEMENTS(*InfoPtr) EQ 0) THEN $
>>>   MESSAGE, 'State information structure is undefined'

```
>>>    IF ( KEYWORD_SET( No_Copy ) ) THEN BEGIN
>>>      Info = TEMPORARY( *InfoPtr )
>>>    ENDIF ELSE BEGIN
>>>      Info = *InfoPtr
>>>    ENDELSE
>>>    IF ( Info.Debug EQ 1 ) THEN PRINT, 'GetState'
>>>  END
>>>  and to set the info state
>>>  ; Routine to set the Info state
>>>  PRO SetState, ID, Info, No_Copy = No_Copy
>>>    ; -- Get pointer
>>>    WIDGET_CONTROL, ID, GET_UVALUE = InfoPtr
>>>    IF ( PTR_VALID( InfoPtr ) EQ 0 ) THEN $
>>>    MESSAGE, 'State information pointer is invalid'
>>>    ; -- Set state information structure
>>>    IF ( N_ELEMENTS( Info ) EQ 0 ) THEN $
>>>      MESSAGE, 'State information structure is undefined'
>>>    IF ( KEYWORD_SET( No_Copy ) ) THEN BEGIN
>>>      *InfoPtr = TEMPORARY( Info )
>>>    ENDIF ELSE BEGIN
>>>      *InfoPtr = Info
>>>    ENDELSE
>>>    IF ( (*InfoPtr).Debug EQ 1 ) THEN PRINT, 'SetState'
>>>  END
>>>  My widget event handlers then do something like:
>>>  FUNCTION ComponentTest_LogLin_Event, Event
>>>    ; -- Get main info state
>>>    GetState, Event.Top, Info
>>>    ; -- Print debug statement if required
>>>    IF ( Info.Debug EQ 1 ) THEN PRINT, 'ComponentTest_LogLin_Event'
>>>    ; -- Set the selected variable number index
>>>    Info.LogLin_Index = Event.Value
>>>    ; -- Save info state
>>>    SetState, Event.Top, Info
>>>    ; -- Display the result
>>>    ComponentTest_Display, Event.Top
>>>    RETURN, 0
>>>  END
>>>  Note how I call GetState and then SetState. Because I tend not to use /no_copy, it's
>>>  really only an issue when I update a component of the info state (like in my example
>>>  above). But, if you *do* use /no_copy, then I think you have to call SetState again to
>>>  replace the info pointer.
>>>  cheers,
>>>  paulv
>>  Thanks Justus and Paulv
>>  You help me very much. And it was very useful.
>>  Cheers
>>  Dave
```

>
> Hi Justus
> I encounter a now problem:
> Say I have set and get 2 arrays by this method:
> widget_control, event.top, set_uvalue=A, /nocopy
> ï¿½
> widget_control, event.top, get_uvalue=A, /nocopy
>
> and from another button:
> widget_control, event.top, set_uvalue=B, /nocopy
> ï¿½
> widget_control, event.top, get_uvalue=B, /nocopy
>
>
>
> Now I want to get both A&B in another button like this:
>
>
> widget_control, event.top, get_uvalue=A, /nocopy
> widget_control, event.top, get_uvalue=B, /nocopy
>
> But widget just accepts one array. How I can take this two arrays
> simultaneously in another button?

Don't store A and B separately. Store them both. When you create the widget hierarchy, do something like,

```
Info = {A:PTR_NEW(/ALLOCATE_HEAP), $
        B:PTR_NEW(/ALLOCATE_HEAP)  }
InfoPtr = PTR_NEW( Info )
WIDGET_CONTROL, Top_Level_Base_ID, SET_UVALUE = InfoPtr
```

This assumes you don't know the sizes of A and B ahead of time, hence the PTR_NEW(/ALLOCATE_HEAP). Also, if you store a structure, then you can easily add more data to your widget info state as your add more functionality to your GUI application, but without breaking existing stuff.

Then when you need to store them in an event handler:

```
  GetState, Event.Top, Info
  *Info.A = array_with_Adata
  *Info.B = array_with_Bdata
  SetState, Event.Top, Info
```

and also get them both later on:

```
  GetState, Event.Top, Info
  X = *Info.A
```

Y = *Info.B

This type of thing is covered frequently in both Liam's and David's books - as well as numerous examples on David's websites. FWIW my methodology is shamelessly nicked (with some minor alterations) from Liam's book section 9.5 "A GUI Application".

cheers,

paulv

Subject: Re: widget_problem
Posted by Vince Hradil on Wed, 23 Jul 2008 14:41:17 GMT
View Forum Message <> Reply to Message

On Jul 23, 9:01 am, d.po...@gmail.com wrote:
> On Jul 22, 5:52 pm, d.po...@gmail.com wrote:
>
>
>
>> On Jul 17, 8:32 pm, Paul van Delst <Paul.vanDe...@noaa.gov> wrote:
>
>>> Justus Skorps wrote:
>>>> > Justus
>>>> > stil can not fix it. i have Liam E.Gumley's book but .....
>>>> > Cheers
>>>> > Dave
>
>>>> after u load your arrays (lets call them A) store them with
>
>>>> widget_control, event.top, set_uvalue=A, /nocopy
>
>>>> You have to change 'event.top' that it fits your program...
>>>> In your second button you can now load the arrays with
>
>>>> widget_control, event.top, get_uvalue=A, /nocopy
>
>>> but don't forget to put them back into your uvalue when you're done!
>
>>>> It is useful to
>
>>>> -store the data in the main widget
>>>> -use a structure to store every data you want
>
>>> I also tend to use procedures to get the Info state:
>
>>> ; Routine to get the Info state
>>> PRO GetState, ID, Info, No_Copy = No_Copy

```
>>>    ; -- Get pointer
>>>    WIDGET_CONTROL, ID, GET_UVALUE = InfoPtr
>>>    IF ( PTR_VALID( InfoPtr ) EQ 0 ) THEN $
>>>      MESSAGE, 'State Information pointer is invalid'
>
>>>    ; -- Get state information structure
>>>    IF ( N_ELEMENTS( *InfoPtr ) EQ 0 ) THEN $
>>>      MESSAGE, 'State information structure is undefined'
>>>    IF ( KEYWORD_SET( No_Copy ) ) THEN BEGIN
>>>      Info = TEMPORARY( *InfoPtr )
>>>    ENDIF ELSE BEGIN
>>>      Info = *InfoPtr
>>>    ENDELSE
>>>    IF ( Info.Debug EQ 1 ) THEN PRINT, 'GetState'
>>>  END
>
>>>  and to set the info state
>
>>>  ; Routine to set the Info state
>>>  PRO SetState, ID, Info, No_Copy = No_Copy
>>>    ; -- Get pointer
>>>    WIDGET_CONTROL, ID, GET_UVALUE = InfoPtr
>>>    IF ( PTR_VALID( InfoPtr ) EQ 0 ) THEN $
>>>    MESSAGE, 'State information pointer is invalid'
>
>>>    ; -- Set state information structure
>>>    IF ( N_ELEMENTS( Info ) EQ 0 ) THEN $
>>>      MESSAGE, 'State information structure is undefined'
>>>    IF ( KEYWORD_SET( No_Copy ) ) THEN BEGIN
>>>      *InfoPtr = TEMPORARY( Info )
>>>    ENDIF ELSE BEGIN
>>>      *InfoPtr = Info
>>>    ENDELSE
>>>    IF ( (*InfoPtr).Debug EQ 1 ) THEN PRINT, 'SetState'
>>>  END
>
>>>  My widget event handlers then do something like:
>
>>>  FUNCTION ComponentTest_LogLin_Event, Event
>>>    ; -- Get main info state
>>>    GetState, Event.Top, Info
>
>>>    ; -- Print debug statement if required
>>>    IF ( Info.Debug EQ 1 ) THEN PRINT, 'ComponentTest_LogLin_Event'
>>>    ; -- Set the selected variable number index
>>>    Info.LogLin_Index = Event.Value
>
>>>    ; -- Save info state
```

>>>   SetState, Event.Top, Info
>
>>>   ; -- Display the result
>>>   ComponentTest_Display, Event.Top
>>>   RETURN, 0
>>> END
>
>>> Note how I call GetState and then SetState. Because I tend not to use /no_copy, it's
>>> really only an issue when I update a component of the info state (like in my example
>>> above). But, if you *do* use /no_copy, then I think you have to call SetState again to
>>> replace the info pointer.
>
>>> cheers,
>
>>> paulv
>
>> Thanks Justus and Paulv
>> You help me very much. And it was very useful.
>> Cheers
>> Dave
>
> Hi Justus
> I encounter a now problem:
> Say I have set and get 2 arrays by this method:
> widget_control, event.top, set_uvalue=A, /nocopy
> …
> widget_control, event.top, get_uvalue=A, /nocopy
>
> and from another button:
> widget_control, event.top, set_uvalue=B, /nocopy
> …
> widget_control, event.top, get_uvalue=B, /nocopy
>
> Now I want to get both A&B in another button like this:
>
> widget_control, event.top, get_uvalue=A, /nocopy
> widget_control, event.top, get_uvalue=B, /nocopy
>
> But widget just accepts one array. How I can take this two arrays
> simultaneously in another button?
> Any help?
> Cheers
> Dave

Hmmm... in the "combined" button:
widget_control, event.top, set_uvalue={A:A,B:B}, /nocopy
then
widget_contorl, event.top, get_uvalue=AandB

A = AandB.A
B = AandB.B

But why not just store ALL the "state" information in a structure,
then use a pointer to that structure as your uvalue?

I usually do:
[in main:]
info = { A:A, B:B, ... }
infoptr = ptr_new(info,/no_copy)
widget_control, base, set_uvalue=infoptr, /realize
[then call xmanager]

[Then in the event functions:]
widget_control, event.top, get_uvalue=infoptr
widget_control, event.id, get_uvalue=uval

case uval of
...[all the cases, use and set values in (*infoptr), etc.]
endcase

Subject: Re: widget_problem
Posted by Vince Hradil on Wed, 23 Jul 2008 14:42:04 GMT
View Forum Message <> Reply to Message

On Jul 23, 9:34 am, Paul van Delst <Paul.vanDe...@noaa.gov> wrote:
> d.po...@gmail.com wrote:
>> On Jul 22, 5:52 pm, d.po...@gmail.com wrote:
>>> On Jul 17, 8:32 pm, Paul van Delst <Paul.vanDe...@noaa.gov> wrote:
>
>>>> Justus Skorps wrote:
>>>> >> Justus
>>>> >> stil can not fix it. i have Liam E.Gumley's book but .....
>>>> >> Cheers
>>>> >> Dave
>>>> > after u load your arrays (lets call them A) store them with
>>>> > widget_control, event.top, set_uvalue=A, /nocopy
>>>> > You have to change 'event.top' that it fits your program...
>>>> > In your second button you can now load the arrays with
>>>> > widget_control, event.top, get_uvalue=A, /nocopy
>>>> but don't forget to put them back into your uvalue when you're done!
>>>> > It is useful to
>>>> > -store the data in the main widget
>>>> > -use a structure to store every data you want
>>>> I also tend to use procedures to get the Info state:
>>>> ; Routine to get the Info state
>>>> PRO GetState, ID, Info, No_Copy = No_Copy

```
>>>>    ; -- Get pointer
>>>>    WIDGET_CONTROL, ID, GET_UVALUE = InfoPtr
>>>>    IF ( PTR_VALID( InfoPtr ) EQ 0 ) THEN $
>>>>      MESSAGE, 'State Information pointer is invalid'
>>>>    ; -- Get state information structure
>>>>    IF ( N_ELEMENTS( *InfoPtr ) EQ 0 ) THEN $
>>>>      MESSAGE, 'State information structure is undefined'
>>>>    IF ( KEYWORD_SET( No_Copy ) ) THEN BEGIN
>>>>      Info = TEMPORARY( *InfoPtr )
>>>>    ENDIF ELSE BEGIN
>>>>      Info = *InfoPtr
>>>>    ENDELSE
>>>>    IF ( Info.Debug EQ 1 ) THEN PRINT, 'GetState'
>>>>  END
>>>>  and to set the info state
>>>>  ; Routine to set the Info state
>>>>  PRO SetState, ID, Info, No_Copy = No_Copy
>>>>    ; -- Get pointer
>>>>    WIDGET_CONTROL, ID, GET_UVALUE = InfoPtr
>>>>    IF ( PTR_VALID( InfoPtr ) EQ 0 ) THEN $
>>>>    MESSAGE, 'State information pointer is invalid'
>>>>    ; -- Set state information structure
>>>>    IF ( N_ELEMENTS( Info ) EQ 0 ) THEN $
>>>>      MESSAGE, 'State information structure is undefined'
>>>>    IF ( KEYWORD_SET( No_Copy ) ) THEN BEGIN
>>>>      *InfoPtr = TEMPORARY( Info )
>>>>    ENDIF ELSE BEGIN
>>>>      *InfoPtr = Info
>>>>    ENDELSE
>>>>    IF ( (*InfoPtr).Debug EQ 1 ) THEN PRINT, 'SetState'
>>>>  END
>>>>  My widget event handlers then do something like:
>>>>  FUNCTION ComponentTest_LogLin_Event, Event
>>>>    ; -- Get main info state
>>>>    GetState, Event.Top, Info
>>>>    ; -- Print debug statement if required
>>>>    IF ( Info.Debug EQ 1 ) THEN PRINT, 'ComponentTest_LogLin_Event'
>>>>    ; -- Set the selected variable number index
>>>>    Info.LogLin_Index = Event.Value
>>>>    ; -- Save info state
>>>>    SetState, Event.Top, Info
>>>>    ; -- Display the result
>>>>    ComponentTest_Display, Event.Top
>>>>    RETURN, 0
>>>>  END
>>>>  Note how I call GetState and then SetState. Because I tend not to use /no_copy, it's
>>>>  really only an issue when I update a component of the info state (like in my example
>>>>  above). But, if you *do* use /no_copy, then I think you have to call SetState again to
```

>>>> replace the info pointer.
>>>> cheers,
>>>> paulv
>>> Thanks Justus and Paulv
>>> You help me very much. And it was very useful.
>>> Cheers
>>> Dave
>
>> Hi Justus
>> I encounter a now problem:
>> Say I have set and get 2 arrays by this method:
>> widget_control, event.top, set_uvalue=A, /nocopy
>> …
>> widget_control, event.top, get_uvalue=A, /nocopy
>
>> and from another button:
>> widget_control, event.top, set_uvalue=B, /nocopy
>> …
>> widget_control, event.top, get_uvalue=B, /nocopy
>
>> Now I want to get both A&B in another button like this:
>
>>   widget_control, event.top, get_uvalue=A, /nocopy
>> widget_control, event.top, get_uvalue=B, /nocopy
>
>> But widget just accepts one array. How I can take this two arrays
>> simultaneously in another button?
>
> Don't store A and B separately. Store them both. When you create the widget hierarchy, do
> something like,
>
> Info = {A:PTR_NEW(/ALLOCATE_HEAP), $
>   B:PTR_NEW(/ALLOCATE_HEAP) }
> InfoPtr = PTR_NEW(Info)
> WIDGET_CONTROL, Top_Level_Base_ID, SET_UVALUE = InfoPtr
>
> This assumes you don't know the sizes of A and B ahead of time, hence the
> PTR_NEW(/ALLOCATE_HEAP). Also, if you store a structure, then you can easily add more
data
> to your widget info state as your add more functionality to your GUI application, but
> without breaking existing stuff.
>
> Then when you need to store them in an event handler:
>
>   GetState, Event.Top, Info
>   *Info.A = array_with_Adata
>   *Info.B = array_with_Bdata
>   SetState, Event.Top, Info

>
> and also get them both later on:
>
> GetState, Event.Top, Info
> X = *Info.A
> Y = *Info.B
>
> This type of thing is covered frequently in both Liam's and David's books - as well as
> numerous examples on David's websites. FWIW my methodology is shamelessly nicked (with
> some minor alterations) from Liam's book section 9.5 "A GUI Application".
>
> cheers,
>
> paulv

Darn... you type faster than I...

Subject: Re: widget_problem
Posted by David Fanning on Wed, 23 Jul 2008 14:54:07 GMT
View Forum Message <> Reply to Message

Vince Hradil writes:

> Darn... you type faster than I...

My newsreader give you a time stamp of 1.43 AM. That
could be a factor. :-)

Cheers,

David

P.S. Drink more coffee.
--
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: widget_problem
Posted by d.poreh on Wed, 23 Jul 2008 15:12:13 GMT
View Forum Message <> Reply to Message

On Jul 23, 4:54 pm, David Fanning <n...@dfanning.com> wrote:
> Vince Hradil writes:

>> Darn... you type faster than I...
>
> My newsreader give you a time stamp of 1.43 AM. That
> could be a factor. :-)
>
> Cheers,
>
> David
>
> P.S. Drink more coffee.
> --
> 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.")

Thanks Vince
perfect.
Cheers
Dave

Subject: Re: widget_problem
Posted by d.poreh on Wed, 23 Jul 2008 16:11:30 GMT
View Forum Message <> Reply to Message

On Jul 23, 5:12 pm, d.po...@gmail.com wrote:
> On Jul 23, 4:54 pm, David Fanning <n...@dfanning.com> wrote:
>
>
>
>> Vince Hradil writes:
>>> Darn... you type faster than I...
>
>> My newsreader give you a time stamp of 1.43 AM. That
>> could be a factor. :-)
>
>> Cheers,
>
>> David
>
>> P.S. Drink more coffee.
>> --
>> 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.")
>

> Thanks Vince
> perfect.
> Cheers
> Dave
Vince

The combined data AandB works very good but in the info and pointer method because I don't know the size of my arrays (in the base (TLB)) the program doesn't accept my structure (because I import my data (arrays) in a button (in the menu bar)) and usually an error arising.
I just want to read 3 arrays and then with another array that comes during the process; make an INFO with 4 array but…

Any help?
Cheers
Dave

Subject: Re: widget_problem
Posted by Vince Hradil on Wed, 23 Jul 2008 16:20:02 GMT
View Forum Message <> Reply to Message

On Jul 23, 11:11 am, d.po...@gmail.com wrote:
> On Jul 23, 5:12 pm, d.po...@gmail.com wrote:
>
>> On Jul 23, 4:54 pm, David Fanning <n...@dfanning.com> wrote:
>
>>> Vince Hradil writes:
>>>> Darn... you type faster than I...
>
>>> My newsreader give you a time stamp of 1.43 AM. That
>>> could be a factor. :-)
>
>>> Cheers,
>
>>> David
>
>>> P.S. Drink more coffee.
>>> --
>>> 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.")
>
>> Thanks Vince
>> perfect.
>> Cheers
>> Dave

>
> Vince
>
> The combined data AandB works very good but in the info and pointer
> method because I don't know the size of my arrays (in the base (TLB))
> the program doesn't accept my structure (because I import my data
> (arrays) in a button (in the menu bar)) and usually an error arising.
> I just want to read 3 arrays and then with another array that comes
> during the process; make an INFO with 4 array but…
>
> Any help?
> Cheers
> Dave

Paul had the idea - a pointer to a structure of pointers.

Subject: Re: widget_problem
Posted by d.poreh on Wed, 23 Jul 2008 17:00:05 GMT
View Forum Message <> Reply to Message

On Jul 23, 6:20 pm, Vince Hradil <hrad...@yahoo.com> wrote:
> On Jul 23, 11:11 am, d.po...@gmail.com wrote:
>
>
>
>> On Jul 23, 5:12 pm, d.po...@gmail.com wrote:
>
>>> On Jul 23, 4:54 pm, David Fanning <n...@dfanning.com> wrote:
>
>>>> Vince Hradil writes:
>>>> > Darn... you type faster than I...
>
>>>> My newsreader give you a time stamp of 1.43 AM. That
>>>> could be a factor. :-)
>
>>>> Cheers,
>
>>>> David
>
>>>> P.S. Drink more coffee.
>>>> --
>>>> 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.")
>
>>> Thanks Vince

>>> perfect.
>>> Cheers
>>> Dave
>
>> Vince
>
>> The combined data AandB works very good but in the info and pointer
>> method because I don't know the size of my arrays (in the base (TLB))
>> the program doesn't accept my structure (because I import my data
>> (arrays) in a button (in the menu bar)) and usually an error arising.
>> I just want to read 3 arrays and then with another array that comes
>> during the process; make an INFO with 4 array but…
>
>> Any help?
>> Cheers
>> Dave
>
> Paul had the idea - a pointer to a structure of pointers.

yes it is works very good. that is exactly what i want.
thanks Paul & Vince
Cheers
Dave
