Subject: Re: problem
Posted by taejon on Thu, 02 Nov 2006 08:17:34 GMT
View Forum Message <> Reply to Message

Hi

I found the error, I wrote 'value' instead of 'title' now it works well
thank you very much again

sven ohmann

taejon wrote:
> Thank you very much for your advise. I introduced the lines in the code
> and made some changes.
> It still does not work in the moment. The problem is like this:
>
> I get the following message code when I run the program (compilation
> OK)
>
> Compiled_module : CW_Filed
> Type conversion error : unable to convert given STRING to LONG
> Detected at EFGCALC1 93/.../efgcalc1.pro
> Compiled Module: Xmanager
>
> I send again the code: Line 93... seems to have nothing to do with the
> problem.
> I get this message quite often...
> The program efgcalc1 calls the eventhandler 'efgcalc1_event' and from
> there 'dismis_event' is called. I still have no access to the pointer
> data.
>
> Some idea ?
>
>
> Regards
>
> Sven Ohmann
>
>
> ;** ******************************
> pro efgcalc1_event, event
> widget_control, event.top, get_uvalue=pstate
> widget_control, event.id, Get_Value = buttonValue
> print, buttonvalue
> case buttonValue of
> 'QuitSofort' : widget_control, event.top, /destroy
> 'OPTIONS' : efgcalc1_options_events, event

```
>          'Load Binaryfile' : LoadBinFile_events, event
>          'Dismis'          : dismisdata_event, event
>
>          'Apply'           : Applydataevent, event
>   endcase
>
> end
>  ;************************************************************ ********************************
> pro applydata_event, event
>
>   print, 'Data applied'
> end
>  ;************************************************************ ********************************
>
> pro LoadBinFile_events, event
> ; Hier kann User die von EVOX bereitgestellte Datei auswaehlen !
>   inputfile = DIALOG_PICKFILE(/READ, FILTER="*.bin")
>
> ; Falls keine Datei ausgewaehlt wurde...
>   if (inputfile eq '') then begin $
>     widget_control, event.top, SET_UVALUE=sState, /No_Copy
>     RETURN
>   endif
>   widget_control, (*pstate).VoxelzahlId, get_value = VoxelzahlId
>   print, "Voxelzahl", (*pstate).voxelzahlId
> end
>  ;************************************************************ ********************************
>
> pro efgcalc1
> ; Hauptprogramm, hier wird die graphische Oberfläche gebastelt
> ; Zuerst das Menue oben (File und Options), durch Menu=1 wird 'File' zu
> einem pulldown-menue
> ; Mit Menu = 1 wird Button zu Pulldownmenu
>
>   BaseId           = widget_base(/row, title=' EFG-Berechnung',
> mbar=menubaseId)
>   FileId           = widget_button(menubaseId, Value = 'File', Menu =
> 1)
>   OptionsId        = widget_button(menubaseId, Value = 'Options', Menu
> = 1)
>   Q_Id             = widget_button(menubaseId, Value = 'Quit', Menu=1)
>
> ; Hier unter dem 'File_Button', Event_pro gibt den Eventhandler an
>   BinaryfileId     = widget_button(fileId,     Value = 'Load
> Binaryfile', Event_Pro=efgcalc1_LoadBinFile_events)
>   AsciifileId      = widget_button(fileId,     Value = 'Load
> Asciifile')
>   CoreBinfileId    = widget_button(fileId,     Value = 'Load Binary
```

```
>  Corefile')
>    CoreAsciifileId   = widget_button(fileId,     Value = 'Load Ascii
>  Corefile')
>
>  ; Hier unter 'Quit-Button'
>    QuitId            = widget_button(Q_Id,       Value = 'QuitSofort')
>
>  ; Hier unter dem 'Optionsbutton'
>    MultiselectId     = widget_button(optionsId,  Value = 'Multiselect')
>    ComputeEFGId      = widget_button(optionsId,  Value = 'Compute the
>  EFG')
>    DrawEFGId         = widget_button(optionsId,  Value = 'Draw the EFG')
>
>    SubbaseId         = widget_base(baseId, /col)
>
>  ; Die Gruppe zum auswaehlen von 'settings' und 'tools'
>  ; wTabselId         = CW_BGROUP(subbaseId, ['Settings', 'Tools'])
>
>  ; Das Feld mit dem Titel 'Voxelzahl' (CW_Field ist fertiges Widget in
>  IDL), sowie Kristalldaten
>    VoxelzahlId       = CW_Field(SubbaseId, Title = 'Voxelzahl', Value =
>  100, xsize =5, /Integer)
>    KristalllabelId   = widget_label(subbaseId, Value='Kristallsystem')
>    ccaId             = CW_Field(SubbaseId, Title = 'a0',    value =
>  4.8195, xsize=5, /float)
>    ccbId             = CW_Field(SubbaseId, Title = 'b0',    value =
>  10.480, xsize=5, /float)
>    cccId             = CW_Field(SubbaseId, Title = 'c0',    value =
>  6.0902, xsize=5, /float)
>    ccalphaId         = CW_Field(SubbaseId, Title = 'alpha', value =
>  90.0,   xsize=5, /float)
>    ccbetaId          = CW_Field(SubbaseId, Title = 'beta',  value =
>  90.0,   xsize=5, /float)
>    ccgammaId         = CW_Field(SubbaseId, Title = 'gamma', value =
>  90.0,   xsize=5, /float)
>
>
>  ; Buttons zum Aufnehmen von Voxelzahl und Kristalldaten
>    dismisId          = widget_button(subbaseId , Value = 'Dismis',
>  uvalue='udismis', xsize=30, ysize=30)
>    applyId           = widget_button(subbaseId,  Value = 'Apply',
>  xsize=30, ysize=30)
>
>  ; Hier das Bild wo die Elektronendichte erscheinen soll
>    DrawbaseId        = widget_base(baseId, /col)
>    printId           = widget_base(drawbaseId, /col)
>    DrawId            = widget_draw(printId, xsize=500, ysize=400)
>
```

```
>  ; Hier die Schieberegler fuer das Rendern sowie Texteingabe der
>  d-Elektronenfile
>    renderbaseId      = widget_base(printId, /row)
>    RenderId          = widget_slider(renderbaseId, Title='render',
>  minimum=0, maximum=200)
>    rendertext        = widget_text(renderbaseId, /editable, ysize=2)
>    renderrefresh     = widget_button(renderbaseId, Value='Refresh')
>
>  ; Hier dasselbe fuer das kugelsymmetrische Fc-File
>    rendercorebaseId  = widget_base(printId, /row)
>    RenderCore        = widget_slider(rendercorebaseId, value='render
>  Core data', minimum=0, maximum=200)
>    rendercoretext    = widget_text(rendercorebaseId, /editable, ysize=2)
>    rendcorerefreshID = widget_button(rendercorebaseId,
>  Value='rendercorerefresh')
>
>
>  ; Hier werden die Infos fuer IDL bereitgestellt. Konzept Siehe Fanning
>  S. 154
>
>    widget_control, baseId, /realize
>    widget_control, drawId, get_value=winvis
>
>    state  = {winvis : winvis, voxelzahlId : voxelzahlID, ccaId : ccaId}
>    pstate = ptr_new(state)
>
>    widget_control, baseId, Set_UValue=pstate
>
>
>  ; Xmanager gibt den Eventhandler an, und welches das Hauptprogramm ist
>    xmanager, 'efgcalc1', baseId, Event_Handler = 'efgcalc1_event'
>  ,/no_block
>
>  end
>   ;;************************************************************ ***************
>
>
>  pro dismisdata_event, event
>    print, 'Data canceled'
>
>    widget_control, event.top, get_uvalue=pstate
>    widget_control, (*pstate).ccaId, set_value='0'
>  ;     case value of
>  ;       'udismis' : widget_control, (*pstate).ccaId, set_value='0'
>  ;     endcase
>  end
>   ;************************************************************ ********************************
>
```

>
>
>
>
> David Fanning wrote:
>> taejon writes:
>>
>>> I have a problem of running this program (I am newbie in IDL...:
>>>
>>> When I run this prog (compiles OK) it should when I press the
>>> 'dismis'-button remove the value in the textfield named 'a0'. But I get
>>> always the error:
>>>
>>> "pointer type required in this context"
>>>
>>> can somebody give some help ?
>>
>> You need to find some way to pass the pointer containing
>> your program "state" from the efgcalc1 program module, where
>> it is created, to the event handlers, where it is needed.
>> Typically, we use the user value of the top-level base
>> to do this. Put the pointer there:
>>
>> Widget_Control, tlb, Set_UValue=pstate
>>
>> And when you need it in your event handlers, retrieve
>> it from there:
>>
>> Widget_Control, event.top, Get_UValue=pstate
>>
>> 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.")