Subject: Re: Filename order: interactive arranging
Posted by amacphee on Tue, 03 Oct 2000 00:33:11 GMT
View Forum Message <> Reply to Message

Thank you Marc, this also looks like a good solution to my problem.
With such good advice I've been receiving on IDL topics I feel obliged
to learn something useful so I can help someone else out!

Many thanks for all your help everyone!

Andrew

In article <39D8715B.F3C4FB29@hotmail.com>,
  Marc Schellens <m_schellens@hotmail.com> wrote:
> amacphee@my-deja.com wrote:
>
>> Hi,
>> A little problem with list sorting:
>> I'm trying to produce an animation from data reduced from several
files.
>> I can now home in on the data (thanks to David Fanning), isolate the
>> subsets of interest from each file and produce an MPEG movie. Great!
>> However, because I simply choose the files in a dialog_pickfile, the
>> order of the files in the animation is determined by the OS's
standard
>> file sort mechanism (I think). I've considered using a listbox to
get
>> the indicies of the selected filenames in order of the users
clicking
>> and use these values to re-arrange the filenames in the array
returned
>> by dialog_pickfile. I have not delved into widget programming yet
>> though...
>>
>> I expect that this may not be too uncommon a thing for people to
want to
>> do, so maybe again there is a solution waiting out there somwhere?
>>
>> Thanks for any advice,
>> Andrew
>>
>> Sent via Deja.com http://www.deja.com/
>> Before you buy.
>
> I have a very simple interactive list sorter. I think it might be
just what
> you are looking for.

```
>  You have to provide an array of strings and the click on two list
elements
>  to swap them.
>  It returns the sorted inidices. Index your array with it and that's
it.
>  It is derived form another very ancient program I wrote, therefore the
>  programing style
>  is not so up to date.
>  Cheers,
>  :-) marc
>
>  Example use:
>
>  IDL> a=['123','abc','456','def']
>  IDL> ix=L_Sorter(a)
>
>  clicking around....
>
>  IDL> print,ix
>         2       3       1       0
>  IDL> print,a[ix]
>  456 def abc 123
>
>  ;; function L_Sorter
>  ;; Let the user sort a list (a string array)
>  ;;
>  ;; variables:
>  ;; theList  Stringarray to sort
>  ;;
>  ;; keywords:
>  ;; group    the Groupleaders ID, should be provided, since this
>  ;;          is a modal     dialog (if not provided, it works
nevertheless,
>  faking a main base)
>  ;; title    The windows title
>  ;;
>  ;; return:
>  ;; the sorted index
>
>  ;; eventhandler for L_SelectfromList
>  pro L_Sorter_EVENT,event
>
>  common L_SorterCommon,sortIx,listID,list,last
>
>  WIDGET_CONTROL,Event.Id,GET_UVALUE=Ev
>
>  CASE Ev OF
>      'OK': BEGIN
```

```
>         widget_control,Event.top,/destroy
>      END
>      'REVERSE': BEGIN
>         sortIx=reverse(sortIx)
>         widget_control,listID,SET_VALUE=list[sortIx]
>         last=-1
>      END
>      'CANCEL': BEGIN
>         widget_control,Event.top,/destroy
>         sortIx=-1
>      END
>      'LIST': BEGIN
>         if last ne -1 then begin
>            tmp=sortIx[last]
>            sortIx[last]=sortIx[event.index]
>            sortIx[event.index]=tmp
>            widget_control,listID,SET_VALUE=list[sortIx]
>            last=-1
>         endif else begin
>            last=event.index
>         endelse
>      END
>   ENDCASE
>   end
>
>   ;; L_SelectFromList main function
>   function L_Sorter,theList,TITLE=Title,$
>                     GROUP=group,INIT=init
>
>   common L_SorterCommon
>
>   ;; set default title if TITLE keyword is not set
>   if( n_elements(TITLE) ne 0) then $
>     actTitle=Title $
>   else $
>     actTitle='Define the sort order'
>
>   actNum=n_elements(theList)
>   list=theList
>
>   ;; limit the number of visible lines
>   actYsize = actNum < 30
>
>   ;; main base
>   if n_elements(group) ne 0 then begin
>      MAIN_Sorter = WIDGET_BASE( GROUP_LEADER=group, /col,
TITLE=actTitle,
>   /modal)
```

```
>  endif else begin
>     RealMAIN_Sorter=widget_base()
>     MAIN_Sorter = WIDGET_BASE( GROUP_LEADER=RealMAIN_Sorter, /col,$
>                           TITLE=actTitle, /modal)
>  endelse
>
>  if n_elements(init) eq 0 then sortIx=indgen(actNum) else sortIx=init
>  last=-1
>
>  ;; the list widget
>  listID=WIDGET_LIST(MAIN_Sorter, VALUE=list[sortIx],YSIZE=actYsize,$
>                  UVALUE='LIST')
>
>  ;; the cancel button
>  bb=widget_base(MAIN_Sorter,/row,XPAD=0,YPAD=0)
>  b = WIDGET_BUTTON( bb, VALUE=' Ok ', UVALUE='OK')
>  b = WIDGET_BUTTON( bb, VALUE='Reverse', UVALUE='REVERSE')
>  b = WIDGET_BUTTON( bb, VALUE='Cancel', UVALUE='CANCEL')
>
>  ;; realize and handle by xmanager
>  WIDGET_CONTROL, MAIN_Sorter, /REALIZE
>  XMANAGER, 'L_Sorter', MAIN_Sorter
>
>  return,sortIx
>  END
>
>
```