Subject: Re: Adding libraries to online help
Posted by David Foster on Mon, 21 Apr 1997 07:00:00 GMT
View Forum Message <> Reply to Message

This is a multi-part message in MIME format.

----------7278777A3E7F
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit

David Foster wrote:
>

Sorry, it's Monday. Here is the attached LHELP.PRO that I forgot
in the previous post!

Dave
>>
>
> One alternative that might cause some controversy: we document our
> routines in .doc files which are located in the same directories as
> the .pro files. Then we use a program I wrote called LHELP.PRO which
> finds all such .doc files and allows you to choose from among them
> and display their contents. Text search capabilities make it easier
> to find topics. Also, you can use it to look at the source code.
>
> This might not be an option for software you download from external
> sources, as you'd have to extract the headers and paste into .doc
> files for each routine, but I thought I'd mention it. I've been flamed
> for this method before, but it is *very* convenient.
>

--

~~~~~~ ~~~~~~~
David S. Foster Univ. of California, San Diego
Programmer/Analyst Brain Image Analysis Laboratory
foster@bial1.ucsd.edu Department of Psychiatry
(619) 622-5892 8950 Via La Jolla Drive, Suite 2200
La Jolla, CA 92037
[UCSD Mail Code 0949]
~~~~~~ ~~~~~~~

----------7278777A3E7F
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
Content-Disposition: inline; filename="lhelp.pro"

```
;
;  LHELP.PRO   1-12-95
;
; This routine allows the user to get help information on local
; routines written as general-purpose utility functions. The user
; enters LHELP, ROUTINE_NAME and the documentation file
; ROUTINE_NAME.DOC is displayed.
;
; Alternatively, if no parameter is given then a list widget is
; displayed, from which the user can choose a documentation help
; file to display.
;
;  10-25-93 DF Use EXPAND_PATH to find .DOC file in !PATH .
;  11-04-93 DF Keyword /ALL to display file listing available routines.
;   5-12-94 DF Add widget interface with list/choose capablility.
;   7-15-94 DF Add text search capability.
;  10-31-94 DF Prevent selection of first element of list from crashing
;              program (its just a title).
;  11-22-94 DF Calling LHELP with command-line argument will invoke
;              LHELP widget with argument as current help file displayed
;              (instead of XDISPLAYFILE as before). Add GROUP keyword
;              for when calling from a widget.


 ;------------------------------------------------------------ ----------------
;   Procedure LHELP_FIND_FILES
;
; Procedure to create lists of help filenames, their corresponding
; names, .pro routines in directories where these help files exist,
; and their corresponding names.
 ;------------------------------------------------------------ ----------------

PRO lhelp_find_files

common lhelp_common, d,help_files, proc_files, help_names, proc_names, text

help_files = ''
proc_files = ''

dir_list = EXPAND_PATH(!PATH,/ARRAY,COUNT=dirs)       ; Find .doc files
FOR i = 0, dirs-1 DO BEGIN
 filespec = dir_list(i) + '/*.doc'
 found = file_find(filespec, files)
 if (found gt 0) then begin
  if (help_files(0) eq '') then help_files = files else $
   help_files = [help_files, files]
  filespec = dir_list(i) + '/*.pro'             ; Find .pro files in
```

```
  found = file_find(filespec, files)            ;  these directories
  if (found gt 0) then begin
   if (proc_files(0) eq '') then proc_files = files else $
    proc_files = [proc_files, files]
  endif
 endif
ENDFOR

if (help_files(0) eq '') then begin
 help_files = 'None found'
 proc_files = help_files
endif

np = n_elements(proc_files)
proc_names = strarr(np)
for i = 0, np-1 do begin
 proc_names(i) = strupcase(filename(proc_files(i)))
 pos = strpos(proc_names(i), '.PRO')
 proc_names(i) = strmid(proc_names(i), 0, pos)
endfor
order = sort(proc_names)
proc_files = proc_files(order)
proc_names = proc_names(order)

nr = n_elements(help_files)
help_names = strarr(nr)
for i = 0, nr-1 do begin
 help_names(i) = strupcase(filename(help_files(i)))
 pos = strpos(help_names(i), '.DOC')
 help_names(i) = strmid(help_names(i), 0, pos)
endfor
order = sort(help_names)
help_files = help_files(order)
help_names = help_names(order)

return
END



 ;------------------------------------------------------------ ----------------
;   Procedure LHELP_POSITION
;
; Procedure to position the text in the text widget.
 ;------------------------------------------------------------ ----------------

PRO lhelp_position
```

```
common lhelp_common, d,help_files, proc_files, help_names, proc_names, text

if (d.cur_line lt d.top_line or d.cur_line gt d.bot_line) then begin
 line = FIX( ( (d.cur_line - (d.ypage_size/2)) > 0 )  $
   < (n_elements(text) - d.ypage_size/2) )
 widget_control, d.page, set_text_top_line=line
 d.top_line = line
 d.bot_line = line + d.ypage_size - 1
endif

return
END


 ;------------------------------------------------------------ ----------------
;   Procedure LHELP_HIGHLIGHT
;
; Procedure to highlight current text found.
 ;------------------------------------------------------------ ----------------

PRO LHELP_HIGHLIGHT

common lhelp_common, d,help_files, proc_files, help_names, proc_names, text

chrs = 0
for i = 0, d.cur_line - 1 do chrs = chrs + strlen(text(i)) + 1
chrs = chrs + d.cur_pos
widget_control, d.page, set_text_select=[chrs, strlen(d.search_text)]

return
END


 ;------------------------------------------------------------ ----------------
;   Function LHELP_SEARCH
;
; Function to search for a text string in the current listing.
 ;------------------------------------------------------------ ----------------

FUNCTION lhelp_search, NEXT=next, PREV=prev

common lhelp_common, d,help_files, proc_files, help_names, proc_names, text


text_lines = n_elements(text)
lines_checked = 0
len = strlen(d.search_text)
```

```
if (keyword_set(NEXT)) then begin
 line = d.cur_line
 last = text_lines - 1
 restart = 0
 incr = 1
endif else if (keyword_set(PREV)) then begin
 line = d.cur_line
 last = 0
 restart = text_lines - 1
 incr = -1
endif

while (1) do begin
 lines_checked = lines_checked + 1          ; Break if not found
 if (lines_checked eq text_lines + 2) then goto, NOT_FOUND
 text_uc = strupcase(text(line))
 search_text_uc = strupcase(d.search_text)
 if (line eq d.cur_line and d.searched ne 0) then begin  ; Repeated searches
  if (keyword_set(NEXT)) then begin                  ;  in same line
   pos = strpos(text_uc, search_text_uc, d.cur_pos)
  endif else if (keyword_set(PREV)) then begin
   text_uc = strmid(text_uc, 0, d.cur_pos - (len + 1))
   pos = strpos_last(text_uc, search_text_uc, 0)
  endif
 endif else begin
  if (keyword_set(NEXT)) then begin
   pos = strpos(text_uc, search_text_uc, 0)
  endif else if (keyword_set(PREV)) then begin
   pos = strpos_last(text_uc, search_text_uc, 0)
  endif
 endelse
 if (pos ge 0) then begin
  d.cur_line = line
  d.cur_pos = pos
  lhelp_position                    ; Reposition text in widget
  lhelp_highlight                   ; Highlight search string found
  d.cur_pos = pos + len                 ; Move past current selection
  goto, FOUND
 endif
 line = line + incr
 if (line eq last + incr) then line = restart       ; Wrap to beginning/end
endwhile

NOT_FOUND:
beep & return, -1

FOUND:
d.searched = 1
```

```
return,0

END


 ;------------------------------------------------------------ ----------------
;   Procedure LHELP_EVENT
;
; Event handler.
 ;------------------------------------------------------------ ----------------

PRO lhelp_event, event

common lhelp_common, d,help_files, proc_files, help_names, proc_names, text

name = strmid(tag_names(event, /structure), 7, 1000)
widget_control, event.id, get_uvalue=value

case (name) of
 "BUTTON": begin
  case (value) of
   "QUIT": widget_control, event.top, /destroy
   "CLOSE": widget_control, event.top, /iconify
   "PROCEDURE":begin
    widget_control, d.list, set_value=proc_names
    widget_control, d.list, set_uvalue=proc_files
    end
   "NEXT": begin
    if (d.search_text eq '' or text(0) eq '') then begin
     beep & return
    endif
    widget_control, d.base, /hourglass
    ret = lhelp_search( /next)
    widget_control, d.base, hourglass=0
    end
   "PREV": begin
    if (d.search_text eq '' or text(0) eq '') then begin
     beep & return
    endif
    widget_control, d.base, /hourglass
    ret = lhelp_search( /prev)
    widget_control, d.base, hourglass=0
    end
   "HELP": begin
    widget_control, d.list, set_value=help_names
    widget_control, d.list, set_uvalue=help_files
    end
   else:
```

```
  endcase
  end
 "LIST": begin
  if (event.index eq 0) then return   ; Cant select first element (label!)
  routine = value(event.index-1)      ; Get routine name
  text = file_strarr(routine)
  widget_control, d.page, set_value=text
  d.searched = 0
  d.cur_line = 0
  d.top_line = 0
  d.bot_line = d.ypage_size - 1
  end
 "TEXT_CH": goto, UPDATE_TEXT
 "TEXT_STR": goto, UPDATE_TEXT
 "TEXT_DEL": begin
  UPDATE_TEXT:
  widget_control, d.text, get_value=txt
  d.search_text = txt(0)
  d.searched = 0
  if (name eq "TEXT_CH") then begin      ; ENTER begins search (next)
   if (event.ch eq 10B) then begin
    if (d.search_text eq '' or text(0) eq '') then begin
     beep & return
    endif
    widget_control, d.base, /hourglass
    ret = lhelp_search( /next)
    widget_control, d.base, hourglass=0
   endif
  endif
  end
 else:
endcase

END


 ;------------------------------------------------------------ ----------------
;   Procedure LHELP
;
; Procedure to display documentation file, or allow user to choose a
; routine from a list.
 ;------------------------------------------------------------ ----------------

PRO lhelp, routine, GROUP=group

common lhelp_common, d,help_files, proc_files, help_names, proc_names, text

d = { base: 0L, $      ; Main widget base ID
```

```
  list: 0L, $      ; Routine widget list ID
  page: 0L, $      ; Info text widget ID
  text: 0L, $      ; Search text widget
  search_text: '', $    ; Search text
  searched: 0, $     ; Has current text been searched (1=yes)
  cur_line: 0, $     ; Current line of text
  cur_pos: 0,  $     ; Current position in current line
  top_line: 0, $     ; Line number at top of text widget
  bot_line: 0, $     ; Line number at bottom
  ext: '.doc', $     ; Do you want '.pro' or '.doc' ?
  max_lines: 5000, $    ; Maximum # lines in file
  xpage_size: 100, $    ; Size of text page
  ypage_size:  40 }


if (keyword_set(GROUP)) then begin
 if (widget_info(group, /valid_id) eq 0) then group = 0
endif else begin
 group = 0
endelse


LHELP_FIND_FILES                        ; Get filename lists
if (help_files(0) eq '') then message, 'No local *.doc files found in !PATH'


; Initialize variables

text = ''
d.bot_line = d.ypage_size - 1
help_names = ['<   Overview   >', help_names]
proc_names = ['<   Overview   >', proc_names]


d.base = WIDGET_BASE(/row, xpad=5, ypad=5, title='Local Help', $
  group_leader=group)
d.list = WIDGET_LIST(d.base, /frame, ysize=d.ypage_size - 10, $
  uvalue=help_files, value=help_names)

right_main = WIDGET_BASE(d.base, xpad=5, ypad=5, /column)
menu = WIDGET_BASE(right_main, xpad = 10, ypad=5, /row, /frame)
quit = WIDGET_BUTTON(menu, value=' Quit ', uvalue='QUIT')
clos = WIDGET_BUTTON(menu, value=' Close ', uvalue='CLOSE')

excl = WIDGET_BASE(menu, /row, xpad=5, ypad=5, /frame, /exclusive)
proc = WIDGET_BUTTON(excl, value='Procedures', uvalue='PROCEDURE')
help = WIDGET_BUTTON(excl, value='Help Files', uvalue='HELP')
```

```
junk = WIDGET_BASE(menu)
sear = WIDGET_BASE(menu, /row, xpad=5, ypad=5, /frame)
prev = WIDGET_BUTTON(sear, value='Prev', uvalue='PREV')
next = WIDGET_BUTTON(sear, value='Next', uvalue='NEXT')
d.text = WIDGET_TEXT(sear, xsize=24, ysize=1, /editable, /all_events)
d.page = WIDGET_TEXT(right_main, xsize=d.xpage_size, ysize=d.ypage_size, $
  /scroll)

widget_control, help, /set_button
widget_control, d.base, /realize

; Help info for only one routine (command-line argument)

if (n_elements(routine) gt 0) then begin
 if (strpos(routine, '.doc') lt 0) then $
  routine = routine + '.doc'
 pos = strpos_last(routine, '/')
 if (pos ge 0) then $
  routine = strmid(routine, pos+1, strlen(routine)-(pos+1))
 file = find_procedure(routine)
 if (file eq '') then begin
  beep
  msg =  'Cannot find help file ' + routine
  message, msg, /continue
  if (keyword_set(GROUP)) then begin
   ret = ask(msg, 'Ok', parent=d.base)
   widget_control, d.base, /destroy
   return
  endif
 endif else begin
  text = file_strarr(file)
  widget_control, d.page, set_value=text
  d.searched = 0
  d.cur_line = 0
  d.top_line = 0
  d.bot_line = d.ypage_size - 1
 endelse
endif

XMANAGER, "lhelp", d.base

return
END

----------7278777A3E7F--
```