
Subject: Re: Path handling in different OSs??
Posted by [thompson](#) on Thu, 03 Feb 2000 08:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

Markus Feldt <mfeldt@mpia-hd.mpg.de> writes:

> Hi out there,

> did anybody bother to write some helpers for handling pathnames under
> different OSs in IDL (something like "path_to_os") or so? Something
> that replaces "/" bei "\" or ":" and appends ":" under MAC if needed? Or
> is there some implemented function in idl which allows accessing
> directory structures under differing OSs? Just asking, if not, I'll
> write it myself...

Markus:

This is what we use. Additional supporting routines can be found in

<ftp://sohoftp.nascom.nasa.gov/solarsoft/gen/idl/>

William Thompson

```
;+
; Project   : SSW
;
; Name      : CONCAT_DIR
;
; Purpose   : To concatenate directory and file names for current os.
;
; Explanation : The given file name is appended to the given directory
;               name with the format appropriate to the current operating
;               system. Can be also used to append two directory names
;
; Use       : IDL> full_name = concat_dir(directory,filename)
;             IDL> pixfile = concat_dir('$DIR_GIS_MODEL','pixels.dat')
;
;             IDL> file = ['f1.dat','f2.dat','f3.dat']
;             IDL> dir = '$DIR_NIS_CAL'
;             IDL> f = concat_dir(dir,file)
;
; Inputs    : DIRECTORY      the directory path (string)
;             FILE           the basic file name and extension (string)
;                               can be an array of filenames or directory
;                               names
;
```

```

; Opt. Inputs : None
;
; Outputs   : The function returns the concatenated string.  If the file
;             input is a string array then the output will be a string
;             array also.
;
; Keywords   : DIR -- If set, the second argument is treated as a directory
;             name instead of a file name (it has no effect if not
;             under VMS system)
;             CHECK -- Check the validity of directory name(s) if set
;             NOTRANSULATE - bypass translation of environmental/logicals
;
; Calls      : CHK_DIR, BELL, BREAK_PATH, OS_FAMILY, GET_LOGENV, ARR2STR,
STR_SEP
;
; Restrictions: Assumes Unix type format if os is not VMS or windows.
;
; Side effects: None
;
; Category   : Utilities, Strings
;
; Prev. Hist. : Yohkoh routine by M. Morrison
;
; Written    : CDS version by C D Pike, RAL, 19/3/93
;
; Modified   :
;             Version 2, Liyun Wang, GSFC/ARC, January 3, 1995
;             Made it capable of concatenating directory names
;             Added keywords CHECK and DIR for output
;             Version 3, William Thompson, GSFC, 3 May 1995
;             Modified so spurious $ characters in front of VMS logical names
;             are ignored.  This makes it easier to port software written for
;             Unix to VMS.
;             Version 4, William Thompson, GSFC, 29 August 1995
;             Modified to use OS_FAMILY.
;
;             Version 5, Samuel Freeland, GSFC, 21-February 1996
;             Merge SLF change to Decode Environmental/Logicals
;             Version 5.1, Samuel Freeland, LPARL, 12-mar-1996
;             restore /NOTRANS keyword
;             Version 5.2, RAS, HSTX, 20-Jun-1996, protect against '..' from concealed
;             directories in VMS
;             Version 5.3 J. Newmark, 03-Jun-1998, changed loops to long integer
;             Version 6, 14-Jan-1999, William Thompson, GSFC
;             Automatically decode environment variables starting with "$" in
;             the Windows.  Treat case where dirname ends in '/' in Windows.
;
; VERSION:

```

```

;      Version 6, 14-Jan-1999
;-
;
;
FUNCTION concat_dir, dirname, filnam, check=check, dir=dir, nottranslate=nottranslate
;
; Check number of parameters
;
;
IF N_PARAMS() LT 2 THEN BEGIN
    PRINT, ' ' & bell
    PRINT, 'Use:  out_string = concat_dir( directory, filename)'
    PRINT, ' '
    RETURN, "
ENDIF
;
; remove leading/trailing blanks
;
;
dir0 = STRTRIM(dirname, 2)
n_dir = N_ELEMENTS(dir0)

; ----- S.L.Freeland, 22-feb-1995 / Decode Logicals/Environment -----
check=keyword_set(check) ; set boolean,s.l.f, 22-feb-1995

envchk=get_logenv(dir0,outenv=outenv) ; translate logical/environ
envs=where(envchk ne "",envcnt)

if envcnt gt 0 then begin
    case 1 of
        keyword_set(nottranslate): dir0(envs)=outenv
        else: dir0(envs)=envchk(envs)
    endcase
endif
; -----
;
;
; act according to operating system
;
;
IF (!version.os EQ 'vms') THEN BEGIN
    i = 0
    wnot3 = strpos( dir0,'...']')
    w2per = (strpos( dir0,'..') ne -1) and (strpos(dir0,'...') eq -1)
    while i lt n_dir DO BEGIN
;
;
; Call BREAK_PATH to make sure that a leading dollar sign is not a problem.
; If more than one directory is returned, then only use the first one. (Note
; that the first entry in the array returned by break_path is always the null
; path.
;
;
dir0(i) = (break_path(dir0(i)))(1)
;
;

```

```

; ----- RAS, 20-jun-1996 / protect against concealed file's period
;
    if wnot3(i) eq -1 then dir0(i) = arr2str( str_sep( dir0(i),'.'],'])
    if w2per(i) then dir0(i) = arr2str( str_sep( dir0(i), '..',')')
    IF check THEN BEGIN
        IF NOT chk_dir(dir0(i)) THEN MESSAGE,/cont,$
            'Warning: directory '+dir0(i)+' does not exist'
    ENDIF
    last = STRMID(dir0(i), STRLEN(dir0(i))-1,1)
    IF ((last NE ']') AND (last NE ':')) THEN BEGIN
        dir0(i) = dir0(i) + ':' ;append an ending ':'
    ENDIF
i = i + 1
ENDwhile
;
; Under Windows, if the directory starts with a dollar sign, then check to see
; the if it's really an environment variable. If it is, then substitute the
; the environment variable for the directory name.
;
    ENDIF ELSE IF OS_FAMILY() EQ 'Windows' THEN BEGIN
        FOR i = 0l, n_dir-1 DO BEGIN
            FIRST = STRMID(DIR0(I), 0, 1)
            IF FIRST EQ '$' THEN BEGIN
                SLASH = STRPOS(DIR0(I)+'/', '/') < STRPOS(DIR0(I)+'\', '\')
                TEST = GETENV(STRMID(DIR0(I),1,SLASH-1))
                IF TEST NE '' THEN BEGIN
                    IF STRLEN(DIR0(I)) GT SLASH THEN TEST = TEST + $
                        STRMID(DIR0(I),SLASH,STRLEN(DIR0(I))-SLASH)
                    DIR0(I) = TEST
                ENDIF
            ENDIF
        ENDIF
;
        IF check THEN BEGIN
            IF NOT chk_dir(dir0(i)) THEN MESSAGE,/cont,$
                'Warning: directory '+dir0(i)+' does not exist'
        ENDIF
        last = STRMID(dir0(i), STRLEN(dir0(i))-1, 1)
        IF (last NE '\') AND (last NE '/') AND (last NE ':') THEN BEGIN
            dir0(i) = dir0(i) + '\' ;append an ending '\'
        ENDIF
    ENDFOR

ENDIF ELSE BEGIN
    FOR i = 0l, n_dir-1 DO BEGIN
        IF check THEN BEGIN
            IF NOT chk_dir(dir0(i)) THEN MESSAGE,/cont,$
                'Warning: directory '+dir0(i)+' does not exist'
            ENDIF
        ENDIF
    ENDFOR

```

```

        IF (STRMID(dir0(i), STRLEN(dir0(i))-1, 1) NE '/') THEN BEGIN
            dir0(i) = dir0(i) + '/' ;append an ending '/'
        ENDIF
    ENDFOR
ENDELSE
;
;
; no '/' needed when using default directory
;
FOR i = 0l, n_dir-1 DO BEGIN
    IF (dirname(i) EQ "") THEN dir0(i) = "
ENDFOR

;-----
; Under Unix and Windows, FILNAM can still be appended to dir0 even if it
; is a directory name. Under VMS, however, we have to check to see if
; FILNAM is a directory name, and if it is, we have to do more to append
; it to dir0.
;-----
IF !version.os EQ 'vms' AND KEYWORD_SET(dir) THEN BEGIN
    dirlen = STRLEN(dir0(0))
    IF STRMID(dir0(0), dirlen-1,1) EQ ':' THEN BEGIN
;-----
;     dir0(0) is a logical dir name; we need to get its real name
;-----
        realdir = chklog(dir0(0))
        IF realdir EQ "" THEN $
            MESSAGE, dir0(0)+' is not a directory!'
        ENDIF ELSE realdir = dir0(0)
        temp = STRMID(realdir,0,STRLEN(realdir)-1)+'.'
        FOR i = 0l, N_ELEMENTS(filnam)-1 DO BEGIN
            new_name = temp+STRUPCASE(filnam(i))+']'
            IF check THEN BEGIN
                IF chk_dir(new_name,outdir,/full) THEN BEGIN
                    IF N_ELEMENTS(result) EQ 0 THEN $
                        result = outdir $
                    ELSE $
                        result = [result, outdir]
                ENDIF ELSE $
                    message, 'Warning: '+new_name+' is not a valid directory name!',$
                        /continue
            ENDIF ELSE BEGIN
                IF N_ELEMENTS(result) EQ 0 THEN $
                    result = new_name $
                ELSE $
                    result = [result, new_name]
            ENDIF ELSE $
            ENDELSE
        ENDFOR
        IF N_ELEMENTS (result) NE 0 THEN RETURN, result ELSE RETURN, "

```

```
ENDIF ELSE RETURN, dir0 + filnam
END
```

Subject: Re: Path handling in different OSs??
Posted by [Martin Schultz](#) on Thu, 03 Feb 2000 08:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

Markus Feldt wrote:

```
>
> Hi out there,
>
> did anybody bother to write some helpers for handling pathnames under
> different OSs in IDL (something like "path_to_os") or so? Something
> that replaces "/" bei "\" or ":" and appends ":" under MAC if needed? Or
> is there some implemented function in idl which allows accessing
> directory structures under differing OSs? Just asking, if not, I'll
> write it myself...
>
> Thanks
>
> Markus
>
```

Take a look at the built-in FILEPATH function. And since it's source code is available ...

Cheers,
Martin

```
--
[ Dr. Martin Schultz  Max-Planck-Institut fuer Meteorologie  [[
[ Bundesstr. 55, 20146 Hamburg  [[
[ phone: +49 40 41173-308  [[
[ fax: +49 40 41173-298  [[
[ martin.schultz@dkrz.de  [[
[
```

Subject: Re: Path handling in different OSs??
Posted by [T Bowers](#) on Thu, 03 Feb 2000 08:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

no, but here's a start if you don't even know how to begin. funny,
i just wrote this last night. it doesn't do what you want, it adds

the passed or current dir to IDL's !path variable.

good luck,
todd b

```
;/!!!!!!!!!!!!!!!!!!!!!!!!!!!!
function tbAddToPath, pathString, APPEND=append, TREE=tree,
WIDGROUPELEADER=widGroupLeader
;+
; NAME:
; tbAddToPath
;
; PURPOSE:
; This function creates a new path string by adding pathString to the IDL
!path var.
;
; CATEGORY:
; Utility.
;
; CALLING SEQUENCE:
; retCode = tbAddToPath(pathString, APPEND=0, TREE=1,
WIDGROUPELEADER=widGroupLeader)
;
; INPUTS:
; pathString: Sting containg the directory to add to the current !path
; IDL path variable. If not supplied, adds the current working directory.
;
; KEYWORDS:
; APPEND: Setting this keyword appends new path info to end of !path.
; By default, adds it to the beginning.
; TREE: Set this keyword to include sub directories of the dir to !path.
; Unfortunately, only sub dirs with .pro or .sav files in them will be
included.
; See IDL docs on paths.
; WIDGROUPELEADER: Set this keyword to the group leader widget id.
;
; OUTPUTS:
; Returns a string with the newly created path, or NULL ("" ) if theres a
problem.
;
; SIDE EFFECTS:
; Displays an error dialog in catch block.
;
; RESTRICTIONS:
; The "Warning" Dialog_Message dialog is used.
;
; USES:
; Error_Message() from DFanning library
```

```

;
;
; EXAMPLE:
; retCode = tbAddToPath(pathString, /append, WIDGROUPLER=widGrpLdr)
;
;
; MODIFICATION HISTORY:
; Written by: Todd Bowers, 02 Feb 2000.
; Modified:
;-

; //Error Handling code////////
catch, errorCode
if (errorCode NE 0) then begin
; //Cancel this error handling block
catch, /cancel
; //IDL's error message
systemAnswer = Error_Message(/Traceback)
; //Application's error message
message, "[ERROR] In function tbAddToPath(). Returning NULL
return, ""
endif
; ///////////////////////////////////////////////////

; ///////////////////////////////////////////////////
; //Check to see if var's/keywords passed. If not, give default values.
if n_elements(pathString) EQ 0 then begin
; //Use current working directory
cd, CURRENT=pathString
endif
; ///////////////////////////////////////////////////

; ///////////////////////////////////////////////////
; //What's proper path delimiter for this OS?
case (strlowcase(!VERSION.OS_FAMILY)) of
"unix" : pathDelim = ":"
"windows" : pathDelim = ";"
"win32" : pathDelim = ";" ;IDL4.0
else : pathDelim = "," ;macOS and VMS
endcase

; //Do you want to add this dir's subdir's too?
if n_elements(tree) EQ 0 then $
addPath = expand_path(pathString, COUNT=dirCount) $
else $
; //Get sub dir's of this dir
addPath = expand_path('+ ' + pathString, COUNT=dirCount)

; //Tokenize the dirs to add
addPathSArr = str_sep(addPath, pathDelim)

```

```

; //Add, but make sure not adding duplicate paths
newPath = !path
currentPathSArr = str_sep(newPath, pathDelim)
for i=0, n_elements(addPathSArr)-1 do begin
  if (where(strlowcase(addPathSArr[i]) EQ strlowcase(currentPathSArr))
EQ -1)[0] then begin
    ; //Add to front or back?
    if n_elements(append) EQ 0 then $
      newPath = addPathSArr[i] + pathDelim + newPath $
    else $
      newPath = newPath + pathDelim + addPathSArr[i]
    endif
  endfor
; //////////////////////////////////////////////////

return, newPath
end; //module tbAddToPath
; //////////////////////////////////////////////////
"Markus Feldt" <mfeldt@mpia-hd.mpg.de> wrote in message
news:389980B4.1620E411@mpia-hd.mpg.de...
> Hi out there,
>
> did anybody bother to write some helpers for handling pathnames under
> different OSs in IDL (something like "path_to_os") or so? Something
> that replaces "/" bei "\" or ":" and appends ":" under MAC if needed? Or
> is there some implemented function in idl which allows accessing
> directory structures under differing OSs? Just asking, if not, I'll
> write it myself...
>
> Thanks
>
> Markus
>
>
>
>
> --
> /| ` \| Markus Feldt          Voice:   +49 6221 528 262
> |_| ` |_| Max-Planck-Institut   Fax:    +49 6221 528 246

> |_____| D-69117 Heidelberg, Germany      Si, !asi es la vida!
>
>
>

```
