You can quickly try this code by pasting it into your LB IDE and running it. The menu that popsup when you right-click in the graphicbox should be a list of all the folders in the LB distribution.
A better test would be to put it in a folder of its own and create subfolders with the names you want to appear on the menu.

This is just one way of making popup menus with the Windows API #user32 dlls. The advantages over the LB popupmenu command are:

If the users changes his mind (I say "his" because women don't do this!) and clicks somewhere else, then the old menu disappears and a new one appears. With the LB command the user has to click one more time.

The main advantage is you can load menus dynamically as they are not "hard coded". So your menus can change to reflect the current state of affairs while the program is running.

```
' program demonstrating API popupmenus - Grahame King Oct 2006
'Based on an idea from Sean Brown
' Released to public domain - September 27th 2006
dim fi$(10,10)
FolderDir$ = DefaultDir$
open "API popups demo" for graphics as #main
#main "trapclose quit.main"
struct popupPoint, x as long, y as long
global popupItems ' number of items in menu
call GetItemsToPopup
#main "when rightButtonUp PopsUpMenu"
wait
end

sub quit.main handle$
   close #handle$
   end
end sub

sub AddMenuItem hndle,MenuRetCode,flag,MenuString$
if flag=0 then flag=_MF_STRING
calldll #user32,"AppendMenuA",_
   hndle as ulong,_
   flag as long,_
   MenuRetCode as long,_
   MenuString$ as ptr,_
   r as long ' return unused
end sub
```

```
sub GetItemsToPopup
files FolderDir$, fi$()
numFiles = val(fi$(0,0))
numFolders = val(fi$(0,1))
' redim and load array
popupItems = numFolders
redim popupArray$(popupItems)
popupArray$(0) = "Folders:}folderMenu" ' header}name of popup
for i = 1 to popupItems
   popupArray$(i) = fi$(i+numFiles,1)
next i
sort popupArray$(), 1 , popupItems
end sub

sub PopsUpMenu Handle$,x,y
' actually pops up menu in popupArray$() at point x,y in window or con
trol, #Handle$
' and converts response to action
calldll #user32,"CreatePopupMenu",hpopup as ulong
heading$ = word$(popupArray$(0),1,"}")
if heading$<>"}" then
   call AddMenuItem hpopup,0,_MF_DISABLED,heading$
   call AddMenuItem hpopup,0,_MF_SEPARATOR,""
end if
flag = 0
for i = 1 to popupItems
   call AddMenuItem hpopup,i,flag,popupArray$(i)
next i
hWnd = hwnd(#Handle$)
popupPoint.x.struct = x
popupPoint.y.struct = y
calldll #user32, "ClientToScreen",_
   hWnd as long,_ ' handle of control or window
   popupPoint as struct,_
'name of struct containing client/screen coorinates
   r as long
tempx = popupPoint.x.struct
tempy = popupPoint.y.struct
flags = _TPM_TOPALIGN OR _TPM_LEFTALIGN OR _TPM_RETURNCMD
CallDLL #user32, "TrackPopupMenu",_
   hpopup as uLong,_
   flags As Long,_
   tempx As Long,_
   tempy As Long,_
   0 As Long,_
   hWnd as uLong,_
```

```
   0 As Long,_
   re as long ' user's selection (because _TPM_RETURNCMD flag set)
   calldll #user32,"DestroyMenu",hpopup as long
'take action on user's selection
popupID$ = word$(popupArray$(0),2,"}")
select case popupID$
   case "folderMenu"
      if re<>0 then
      notice "You selected "+popupArray$(re);resp$
   else
   end if
   ' other menus if you have them could go here

' but you would have to have a way to introduce the menu into popupArr
ay
   case "otherMenu"
   case else
end select
end sub
```