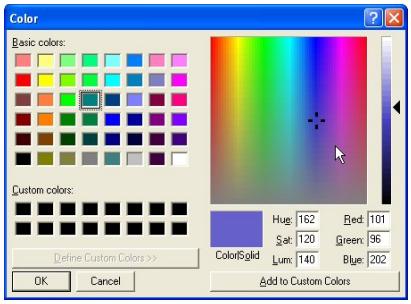
API COLOR DIALOG

- DennisMcK

Liberty BASIC has a perfectly good color dialog built right in. If you need to have something different though, like a color dialog that pops up fully opened, then you need to do a little work and use the common control color dialog. At first it seems like this would be a stright-forward api call but there is a catch or two. The first catch is this dialog does not like LB structures! The dialog will open and display the color you want but when you move the slider or crosshair to select a color it causes LB to crash. A fix for this is to allocate a little memory and copy the LB structure into it, then the dialog can manipulate this memory all it wants to without affecting your program. When the dialog ends the info can be collected and used. The second catch is getting the dialog to open up in the center of the screen. From my own observations the dialog opens with its upper left corner at the upper left corner of the client area of it's owner window. So from that observation a centering trick evolved. A window_popup of 0 width and height can be used as the dialog's owner and positioned where the dialog's upper left corner should be. The size 450 x 325 for the color dialog is an estimate.




```
nomainwin

'this struct is just used for padding
'in this snippet.
struct custClr,_
    x as char[64]

struct cc,_ 'CHOOSECOLOR
    lStructSize as ulong,_
    hwndOwner as ulong ,_
    hInstance as ulong,_
    rgbResult as ulong,_
    lpCustColors as struct,_
    Flags as ulong,_
    lCustData as ulong,_
    lpfnHook as ulong,_
    lpTemplateName$ as ptr
```

```
    WindowWidth = 0
    WindowHeight = 0
    UpperLeftX = Int((DisplayWidth - 450) / 2)
    UpperLeftY = Int((DisplayHeight - 325) / 2)
    open " " for window_popup as #hWndDlgOwner
    hWndDlgOwner = hwnd(#hWndDlgOwner)

    ccSize = len(cc.struct)
    cc.lStructSize.struct = ccSize
    cc.hwndOwner.struct = hWndDlgOwner
    cc.lpCustColors.struct = custClr.struct
    'set the initial color for the dialog to open with.
    cc.rgbResult.struct = 14483455 'RGB(255 255 220)
    cc.Flags.struct = 1 or 2  'CC_FULLOPEN or CC_RGBINIT


'Allocate memory for a CHOOSECOLOR structure and copy the cc.struct to
 it.
    CallDll #kernel32, "GlobalAlloc", _GMEM_FIXED as long,
 ccSize as ulong, hMem as ulong
    CallDll #kernel32, "GlobalLock", hMem as ulong, pMem as long
    CallDll #kernel32,"RtlMoveMemory", pMem as long, cc as struct, _
        ccSize as long, ret as void


'make the call with the memory pointer as the argument for the structu
re.
    calldll #comdlg32, "ChooseColorA", pMem as long, ret as long

    'The choosecolor dialog has returned the selected color in
    'the structure located in memory. To get these values and use them
    'just point the cc.struct to the memory location pMem.
    cc.struct = pMem

    ret = cc.rgbResult.struct
    notice strRGB$(ret)

    close #hWndDlgOwner

    'always free the allocated memory.
    CallDll #kernel32, "GlobalFree", hMem as ulong, ret as long
end

function strRGB$(intColor)
    Blue = int(intColor / (256*256))
```

```
    Green = int((intColor  - Blue *256*256) / 256)
    Red = int(intColor - Blue*256*256 - Green*256)
    strRGB$ = str$(Red)+" "+str$(Green)+" "+str$(Blue)
end function
```