
Project1 - Microsoft Visual Basic [design]
File Edit View Project Format Debug Run Query Diagram Tools Add-Ins
Window Help
General
مدیریت فایل ها و شاخه ها
در این فصل به بررسی تکنیکهای مربوط به کار با فایل‌ها
و شاخه‌ها، که اجزای اصلی ساختار ذخیره اطلاعات در ویندوز
به شمار می روند می پردازیم.
http://www.mazsoft.com

بارکردن یک فایل بزرگ در حافظه به کمک روالهای API

نام فایل	تاریخ	نگارش ویژوال بیسیک	نگارش ویندوز
100513.TXT	۲۱ ژانویه ۱۹۹۵	2 , 3	3.1

در این بخش نشان داده می شود که چگونه یک فایل (یک فایل بزرگ مانند یک فایل گرافیک) را با هر اندازه به درون حافظه بار کرده یا یک بلوک از حافظه را به درون یک فایل ذخیره کنیم. روش گفته شده در این بخش فقط برای نگارش ۳/۱ ویندوز یا بالاتر قابل استفاده خواهد بود.

برای اینکه یک برنامه آزمایشی برای نمایش چگونگی انجام این کار بوجود آورید مراحل زیر را انجام دهید:

۱- ویژوال بیسیک را اجرا نمایید یا از منوی File گزینه New Project را انتخاب نمایید. بدین ترتیب Form1 بصورت پیش فرض بوجود آمده است.

۲- از منوی File گزینه New Module را انتخاب نمایید تا Module1 بوجود بیاید.

۳- لیست زیر را در بخش general و قسمت declarations از Module1 وارد نمایید:

```
' OpenFile() Structure
Type OFSTRUCT
   cBytes As String * 1
   fFixedDisk As String * 1
   nErrCode As Integer
   reserved As String * 4
   szPathName As String * 128
End Type

' OpenFile() Flags
Global Const OF_READ = &H0
Global Const OF_WRITE = &H1
Global Const OF_READWRITE = &H2
Global Const OF_SHARE_COMPAT = &H0
Global Const OF_SHARE_EXCLUSIVE = &H10
Global Const OF_SHARE_DENY_WRITE = &H20
Global Const OF_SHARE_DENY_READ = &H30
Global Const OF_SHARE_DENY_NONE = &H40
Global Const OF_PARSE = &H100
Global Const OF_DELETE = &H200
Global Const OF_VERIFY = &H400
```

```
Global Const OF_CANCEL = &H800
Global Const OF_CREATE = &H1000
Global Const OF_PROMPT = &H2000
Global Const OF_EXIST = &H4000
Global Const OF_REOPEN = &H8000
' Enter each of the following Declare statements on one, single line:
Declare Function OpenFile Lib "Kernel" (ByVal lpFilename As _
   String, lpReOpenBuff As OFSTRUCT, ByVal wStyle As Integer) As Integer
Declare Function hRead Lib "kernel" Alias "_hread" (ByVal hFile As _
   Integer, lpMem As Any, ByVal lSize As Long) As Long
Declare Function hWrite Lib "Kernel" Alias "_hwrite" (ByVal hFile _
   As Integer, lpMem As Any, ByVal lSize As Long) As Long
Declare Function lClose Lib "kernel" Alias "_lclose" (ByVal hFile _
   As Integer) As Integer
' Global Memory Flags
Global Const GMEM_FIXED = &H0
Global Const GMEM_MOVEABLE = &H2
Global Const GMEM_NOCOMPACT = &H10
Global Const GMEM_NODISCARD = &H20
Global Const GMEM_ZEROINIT = &H40
Global Const GMEM_MODIFY = &H80
Global Const GMEM_DISCARDABLE = &H100
Global Const GMEM_NOT_BANKED = &H1000
Global Const GMEM_SHARE = &H2000
Global Const GMEM_DDESHARE = &H2000
Global Const GMEM_NOTIFY = &H4000
Global Const GMEM_LOWER = GMEM_NOT_BANKED
Global Const GHND = (GMEM_MOVEABLE Or GMEM_ZEROINIT)
Global Const GPTR = (GMEM_FIXED Or GMEM_ZEROINIT)

' Enter each of the following Declare statements on one, single line:
Declare Function GlobalAlloc Lib "Kernel" (ByVal wFlags As _
    Integer, ByVal dwBytes As Long) As Integer
Declare Function GlobalLock Lib "Kernel" (ByVal hMem As Integer) _
    As Long
Declare Function GlobalUnlock Lib "Kernel" (ByVal hMem As Integer) _
    As Integer
Declare Function GlobalFree Lib "Kernel" (ByVal hMem As Integer) _
    As Integer
```

۴- لیست زیر را در روال مربوط به واقعه Form_Load از Form1 وارد نمایید:

```
Sub Form_Load ()

   Dim InpFile As String
   Dim OutFile As String
   Dim hFile As Integer
   Dim fileStruct As OFSTRUCT
   Dim FSize As Long
   Dim BytesRead As Long
   Dim BytesWritten As Long
   Dim hMem As Integer
   Dim lpMem As Long
   Dim r As Integer
   Me.Show
   'Insert the name of a bitmap or file that is greater than 64K.
   '256COLOR.BMP is less than 5K in size, however, the routine
```

```
   'below still demonstrates how to read and write a file of any size

   InpFile = "C:\WINDOWS\256COLOR.BMP"
   OutFile = "C:\WINDOWS\TEST.BMP"

   'Get the size of the file to be read
   FSize = FileLen(InpFile)
   If FSize > 0 Then

      'Allocate a block of memory equal to the size of the input file.
      hMem = GlobalAlloc(GPTR, FSize)

      If hMem <> 0 Then
         lpMem = GlobalLock(hMem)

         'Read the file into memory
         hFile = OpenFile(InpFile, fileStruct, OF_READ Or _
               OF_SHARE_DENY_NONE)
         BytesRead = hRead(hFile, ByVal lpMem, FSize)

         MsgBox Format(BytesRead) & " bytes read into memory"

         r = lClose(hFile)

         'Write the file back to disk to verify the file was
         'read correctly
         hFile = OpenFile(OutFile, fileStruct, OF_CREATE Or _
                         OF_WRITE Or OF_SHARE_DENY_NONE)
         BytesWritten = hWrite(hFile, ByVal lpMem, FSize)

         MsgBox Format(BytesWritten) & " bytes written to output file"

         r = lClose(hFile)

         'Free resources
         r = GlobalUnlock(hMem)
         r = GlobalFree(hMem)
      Else
         MsgBox "Not enough memory to store file"
      End If
   Else
      MsgBox "Input file is zero bytes in length"
   End If
   End
End Sub
```

۵- از منوی Run گزینه Start را انتخاب نمایید یا F5 را فشار دهید تا برنامه اجرا شود. Form1 نمایش داده خواهد شد و برنامه به پایان می رسد.

٦- به وسیله PaintBrush یا دیگر برنامه های ویرایش فایلهای گرافیکی فایل C:\WINDOWS\TEST.BMP را بازیابی نمایید تا اطمینان حاصل نمایید که محتویات این فایل همانند فایل C:\WINDOWS\256COLOR.BMP است.

یافتن فایلهای مخفی و سیستم به کمک توابع DIR و GetAttr

نام فایل	تاریخ	نگارش ویژوال بیسیک	نگارش ویندوز
104685.TXT	۲۱ ژانویه ۱۹۹۵	2 , 3	3.1

در این بخش نشان داده می شود که چگونه فایلهای فقط خواندنی[۱]، سیستم[۲] و مخفی[۳] را توسط تابع DIR[$] و GetAttr( ) تعیین نماییم. تابع DIR[$] می تواند دو پارامتر اختیاری نام فایل و خصوصیات را دریافت نماید. اگر پارامتر خصوصیات محتوی برچسب دیسک[۴] باشد تابع Dir[$] بقیه خصوصیات دیگر فایل را نادیده می گیرد. اگر این پارامتر دارای مقدار ATTR_HIDDEN، ATTR_SYSTEM یا ATTR_DIRECTORY باشد تابع DIR[$] فایلهایی که هیچگونه خصوصیت ویژه ای ندارند را نیز باز می گرداند.

درصورتیکه فقط پارامتر نام فایل مورد استفاده قرار گیرد، این تابع بدون توجه به خصوصیت فایل، فایلهایی را بر می گرداند که فقط با پارامتر نام فایل مطابقت نمایند.

برای اینکه فقط فایلهای دارای یک خصوصیت ویژه مانند فقط خواندنی، مخفی یا سیستم را بیابیم باید از تابع Dir[$] به همراه تابع GetAttr( ) استفاده نماییم. مثال زیر نشان می دهد که چگونه فایلهای فقط مخفی (فایلهایی که خصوصیت مخفی ATTR_HIDDEN را به تنهایی یا با ATTR_ARCHIVE دارا باشند) را تعیین کنیم.

برای ایجاد این مثال مراحل زیر را قدم به قدم انجام دهید:

۱- ویژوال بیسیک را اجرا نمایید یا از منوی File گزینه New Project را انتخاب نمایید. بدین ترتیب Form1 بصورت پیش فرض بوجود آمده است.

۲- لیست زیر را در قسمت declarations مربوط به Form1 وارد نمایید:

---

[۱] Read Only

[۲] System

[۳] Hidden

[۴] Disk Label

```
Const ATTR_NORMAL = 0
Const ATTR_READONLY = 1
Const ATTR_HIDDEN = 2
Const ATTR_SYSTEM = 4
Const ATTR_VOLUME = 8
Const ATTR_DIRECTORY = 16
Const ATTR_ARCHIVE = 32
```

۳- یک List Box و یک Command Button به Form1 اضافه کنید.

۴- لیست زیر را در واقعه Click مربوط به Command Button وارد نمایید:

```
Sub Command1_Click ()
   Dim filename As String
   Dim attr As Integer
   ' retrieve hidden and normal files
   filename = Dir$("c:\", ATTR_HIDDEN)
   Do Until filename = ""
      attr = GetAttr("c:\" & filename)
      ' if the file has the hidden attribute
      If (attr And ATTR_HIDDEN) Then
         ' select it
         List1.AddItem filename
      End If
      filename = Dir$
   Loop
End Sub
```

۵- برنامه را اجرا نموده و روی Command Button کلیک نمایید تا فایلهای مخفی موجود در شاخه اصلی درایو خود را مشاهده نمایید.

انتخاب بیش از یک فایل توسط ابزار دیالوگهای عمومی[۵]

نام فایل	تاریخ	نگارش ویژوال بیسیک	نگارش ویندوز
145612.TXT	۷ آگوست ۱۹۹۸	4 , 5	95 / 98 / NT

ابزار دیالوگهای عمومی مربوط به کار با فایلها این امکان را به برنامه نویس می دهد که بتواند یک فایل یا بیشتر از یک فایل را بطور هم زمان در اختیار برنامه های ویژوال بیسیک

[۵] Common Dialog Control

قرار دهد. ابزار دیالوگهای عمومی فایلها این امکان را به شما می دهد تا از درون برنامه خود به ساختمان فایلها و دایرکتوریهای موجود روی هارد دیسک دسترسی یابید. برای مثال، اگر کاربر برنامه شما بخواهد یک فایل متن را انتخاب نماید، می توانید یک دیالوگ فایل را نمایش دهید تا کاربر بتواند در دایرکتوریها به دنبال فایل مورد نظر خود بگردد. با تنظیم خصوصیت Flags مربوط به ابزار دیالوگهای عمومی و قراردادن مقدار OFN_ALLOWMULTISELECT در آن، کاربر شما خواهد توانست چندین فایل را بطور هم زمان انتخاب نماید. با نگاه داشتن کلید SHIFT یا CTRL و کلیک کردن روی نام فایلها می توان چندین فایل را هم زمان انتخاب نمود.

زمانی که برنامه ویژوال بیسیک بخواهد روی فایلهای انتخاب شده توسط کاربر پردازشی انجام دهد، لازمست ابتدا نام تک تک فایلها را بطور مجزا از خصوصیت Filename بیرون بکشد. تمامی نام فایلهای انتخاب شده توسط کاربر بصورت یک رشته حرفی بلند در این خصوصیت ذخیره شده اند. هر نام فایل توسط یک فاصله (کاراکتر کد ۳۲) از نام فایل بعدی جدا شده است.

بنابراین، برای بدست آوردن هر نام فایل لازمست از تابع Instr استفاده نمایید تا محل فاصله های جداکننده را بیابید. تابع Instr مکان کاراکتر فاصله را در Filename بر می گرداند. در مرحله بعد باید توسط تابع Mid هر نام فایل را از میان بقیه بیرون بکشید.

مثال زیر نشان می دهد که چگونه نام فایلهای انتخاب شده توسط دیالوگ فایل را بدست آوریم:

۱- یک پروژه جدید در ویژوال بیسیک بوجود آورید. Form1 بطور پیش فرض بوجود می آید.
۲- یک ابزار دیالوگ به Form1 اضافه نمایید. CommonDialog1 بطور پیش فرض بوجود می‌آید.
۳- یک ابزار Text Box به Form1 اضافه نمایید. Text1 بطور پیش فرض بوجود می آید.
۴- یک ابزار Text Box دیگر به Form1 اضافه نمایید. Text2 بطور پیش فرض بوجود می آید.
۵- یک Command Button به Form1 اضافه نمایید. Command1 نام پیش فرض آن خواهد بود.

۶- لیست زیر را به واقعه Click مربوط به Command1 اضافه نمایید:

```
Private Sub Command1_Click()
    Dim I As Integer
    Dim Y As Integer
    Dim Z As Integer
    Dim FileNames$()

    CommonDialog1.filename = ""
    CommonDialog1.Filter = "All Files|*.*"
    CommonDialog1.Flags = cdlOFNAllowMultiselect
    CommonDialog1.Action = 1
    CommonDialog1.filename = CommonDialog1.filename & Chr(32)

    Z = 1
    For I = 1 To Len(CommonDialog1.filename)
        I = InStr(Z, CommonDialog1.filename, Chr(32))
        If I = 0 Then Exit For
        ReDim Preserve FileNames(Y)
        FileNames(Y) = Mid(CommonDialog1.filename, Z, I - Z)
        Z = I + 1
        Y = Y + 1
    Next
    If Y = 1 Then
        Text1.Text = FileNames(0)
    Else
        Text2.Text = ""
        For I = 0 To Y - 1
            If I = 0 Then
                Text1.Text = FileNames(I)
            Else
                Text2.Text = Text2.Text & UCase(FileNames(I)) & _
                Chr$(13) & Chr$(10)
            End If
        Next
    End If
End Sub
```

۷- برنامه را با فشار F5 اجرا نمایید. روی Command Button کلیک نمایید. پنجره دیالوگ فایل روی صفحه نمایش داده می شود. چند فایل را در لیست فایلها انتخاب نمایید. سپس دکمه OK را فشار دهید. نام فایلها در Text Box دوم نمایش داده خواهد شد و نام دایرکتوری در Text Box اول دیده می شود.

تعیین میزان فضای خالی دیسک

نام فایل	تاریخ	نگارش ویژوال بیسیک	نگارش ویندوز
147686.TXT	۹ می ۱۹۹۷	4	All

در این بخش خواهیم دید که چگونه توسط فراخوانی تابع DiskSpaceFree که درون STKIT416.DLL و STKIT432.DLL قرار دارد، میزان فضای خالی دیسک باقی مانده روی درایو جاری را تعیین نماییم. این فایلهای کتابخانه ای به همراه برنامه Seutp Kit مخصوص ویژوال بیسیک ۴ در اختیار شما قرار می گیرند. این تابع برای پیدا کردن فضای خالی روی درایو قابل استفاده است. اگر از تابع DiskSpaceFree در برنامه های ویژوال بیسیک خود استفاده می نمایید لازم است یکی از فایلهای STKIT416.DLL و STKIT432.DLL را بنا به مورد به همراه برنامه خود در اختیار کاربر قرار دهید.

برای تعیین میزان فضای خالی روی درایو یک تابع دیگر به نام GetDiskFreeSpace نیز در API وجود دارد. این تابع که توسط سیستم عامل Win32 در اختیار گذاشته می شود، اطلاعاتی شامل فضای خالی در مورد یک درایو بخصوص ارائه می دهد. برای نحوه استفاده از این تابع به روش دوم در مثال زیر رجوع کنید. در کتابخانه های STKIT416.DLL و STKIT432.DLL توابع دیگری نیز موجود است که ممکن است مفید باشد. مثلاً برای بدست آوردن اطلاعات واحد تخصیص دیسک[6] مربوط به درایو جاری می توانید از تابع AllocUnit استفاده نموده یا برای تنظیم تاریخ و زمان فایل، تابع SetTime را بکار برید.

مثالی که نشان دهنده نحوه استفاده از این توابع است به عنوان بخشی از ابزارهای ساخت دیسکت نصب در فایل \VB\SETUPKIT\SETUP1\SETUP1.VBP نیز قرار دارد.

مثال زیر نشان می دهد که چگونه میزان فضای خالی روی درایو C را تعیین کنیم:

۱- یک پروژه جدید در ویژوال بیسیک بوجود آورید. Form1 به طور پیش فرض بوجود می آید.

---

[6] Disk Allocation Unit

۲- دو Label (به نامهای Label1 و Label2) و یک Command Button (Command1) روی Form1 بوجود آورید.

۳- خصوصیات ابزارهایی را که در مرحله ۲ بوجود آوردید مطابق جدول زیر تنظیم نمایید:

نام ابزار	خصوصیت	مقدار جدید
Command1	Caption	Press for Free Space on Drive C
Label1	AutoSize	True
Label2	AutoSize	True

۴- در بخش declarations از Form1 لیست زیر را وارد نمایید:

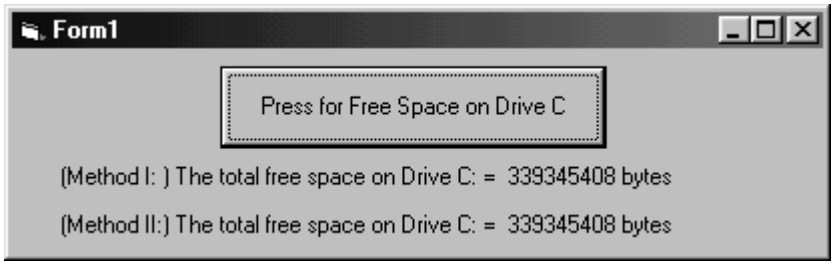
```
#If Win32 Then
   Private Declare Function DiskSpaceFree Lib "STKIT432.DLL" Alias _
      "DISKSPACEFREE" () As Long
   Private Declare Function GetDiskFreeSpace Lib "kernel32" Alias _
      "GetDiskFreeSpaceA" ( ByVal lpRootPathName As String, _
      lpSectorsPerCluster As Long, lpBytesPerSector As Long, _
      lpNumberOfFreeClusters As Long, lpTotalNumberOfClusters As Long) _
      As Long
#Else
   Private Declare Function DiskSpaceFree Lib "STKIT416.DLL" () As Long
#End If
```

۵- لیست زیر را در واقعه Click از Command1 وارد نمایید:

```
Private Sub Command1_Click()
   Dim free_Space As Long
   ChDrive "C:"
   ' Method I (using the Win16 API).
   free_Space = DiskSpaceFree()
   Label1.Caption = "(Method I: ) The total free space on Drive C: = " _
   & Str$(free_Space) & " bytes"
   ' Method II (using the Win32 API).
   #If Win32 Then
      Dim numSectorsPerCluster As Long
      Dim numBytesPerSector As Long
      Dim numFreeClusters As Long
      Dim numTotalClusters As Long
      Dim success As Boolean
      success = GetDiskFreeSpace("C:\", numSectorsPerCluster, _
      numBytesPerSector,  numFreeClusters, numTotalClusters)
      free_Space = numSectorsPerCluster * numBytesPerSector * _
      numFreeClusters
      Label2.Caption = _
      "(Method II:) The total free space on Drive C: = " & _
      Str$(free_Space) & " bytes"
   #Else
      Label2.Caption = "Use Method I for Win16 applications"
```

```
   #End If
End Sub
```

۶- از منوی Run گزینه Start را انتخاب کرده، یا کلید F5 را فشار دهید تا برنامه اجرا شود. برای اینکه میزان فضای خالی درایو C را مشاهده نمایید روی دکمه Command1 کلیک نمایید.



شکل 1 - نتیجه مثال مربوط به تعیین فضای خالی دیسک

استفاده از تابع کپی فایل در ویندوز ۹۸/۹۵			
نام فایل	تاریخ	نگارش ویژوال بیسیک	نگارش ویندوز
151799.TXT	۲۸ آگوست ۱۹۹۸	4 , 5 , 6	95 / 98

API ویندوز این امکان را در اختیار می گذارد که عملیات کپی، جابجایی، تغییر نام یا پاک کردن فایل توسط تابع SHFileOperation موجود در Shell32.dll انجام گیرد. در این بخش خواهیم دید که چگونه لیستی از فایلها را به درون یک پوشه کپی نماییم و هم زمان یک پنجره دیالوگ دارای انیمیشن که میزان پیشرفت عملیات کپی را نشان می دهد را نیز نمایش دهیم.

شکل 2- نمونه ای از پنجره کپی فایل

تابع SHFileOperation بر اساس پارامترهایی که برایش ارسال شود امکان کپی، جابجایی، تغییر نام یا حذف فایلها را فراهم می کند. این تابع به آدرس محل ذخیره ساختمان داده SHFILEOPSSTRUCT نیازدارد. ساختمان داده مذکور از اجزای زیر تشکیل شده است:

- hwnd - اشاره گر پنجره مربوط به پنجره ای که باید اطلاعات مربوط به وضعیت عملیات را نمایش دهد.
- wFunc : کد عملیاتی که باید انجام شود، که یکی از مقادیر زیر است:
  - FO_COPY : فایلهای مشخص شده در pFrom را به محل مشخص شده توسط pTo کپی می کند.
  - FO_DELETE : فایلهای مشخص شده در pFrom را پاک می کند (pTo استفاده نمی شود).
  - FO_MOVE : فایلهای مشخص شده در pFrom را به محل مشخص شده توسط pTo منتقل می کند.
  - FO_RENAME : فایلهای مشخص شده در pFrom را تغییر نام می دهد.
- pFrom : آدرس بافری که مشخص کننده یک یا چند نام فایل باشد. در صورتیکه چند نام فایل را در آن ذخیره می کنیم، باید این نام فایلها توسط کاراکتر تهی (کد صفر) از یکدیگر جداشده باشند. این لیست شامل هر تعداد نام فایل باشد باید توسط دو کاراکتر تهی خاتمه یافته باشد.

- pTo : آدرس بافری که مشخص کننده نام فایل یا دایرکتوری مقصد باشد. در صورتیکه پارامترfFlags حاوی مقدار FOF_MULTIDESTFILES باشد این بافر می تواند محتوی نام فایلهای متعدد باشد. در صورتیکه چند نام فایل را در آن ذخیره می کنیم باید این نام فایلها توسط کاراکتر تهی (کد صفر) از یکدیگر جداشده باشند. این لیست شامل هر تعداد نام فایل باشد باید توسط دو کاراکتر تهی خاتمه یافته باشد.
- fFlags : نشان دهنده علائمی است که نحوه انجام عملیات را کنترل می کنند. این پارامتر می تواند ترکیبی از مقادیر زیر باشد:
  - FOF_ALLOWUNDO : اطلاعات مربوط به بازگرداندن به حالت اولیه را درصورت امکان ذخیره می کند.
  - FOF_CONFIRMMOUSE : در حال حاضر مورد استفاده نیست.
  - FOF_FILESONLY : در صورتیکه نام فایل کلی (*.*) ذکر شده باشد، عملیات مورد نظر را فقط روی فایلها انجام می دهد.
  - FOF_MULTIDESTFILES : مشخص می کند که پارامتر pTo به جای اینکه نشان دهنده یک نام دایرکتوری برای ذخیره فایلهای مقصد باشد، حاوی چند نام فایل مقصد می باشد.
  - FOF_NOCONFIRMATION : به تمامی سئوالات سیستم پاسخ Yes to All خواهد داد.
  - FOF_NOCONFIRMMKDIR : در صورتیکه عملیات مورد نظر نیاز به ایجاد دایرکتوری جدید داشته باشد برای ساخت آن درخواست اجازه نکرده و بطور خودکار عمل خواهد نمود.
  - FOF_NOERRORUI : در صورتیکه مشکلی پیش بیاید هیچ پیغامی روی صفحه نمایش نخواهد داد.
  - FOF_RENAMEONCOLLISION : هنگام کپی، تغییر محل یا تغییر نام فایل، در صورتیکه فایلی با نام مورد نظر در محل مقصد وجود داشته باشد، بطور خودکار نام جدیدی برای فایل مقصد در نظر می گیرد.

- FOF_SILENT : پنجره ای برای نمایش میزان پیشرفت عملیات ظاهر نخواهد شد.
- FOF_SIMPLEPROGRESS : پنجره ای برای نمایش میزان پیشرفت عملیات ظاهر می شود ولی نام فایلها را نشان نمی دهد.
- FOF_WANTMAPPINGHANDLE : در صورتیکه FOF_RENAMEONCOLLISION استفاده شده باشد، پارامتر hNameMappings از اطلاعات مربوط به فایلهای تغییر نام یافته پر خواهد شد.

❑ fAnyOperationsAborted : در صورتیکه کاربر عملیات را متوقف نماید، مقدار TRUE در این پارامتر قرار می گیرد. در غیر اینصورت مقدار این پارامتر FALSE خواهد بود.

❑ hNameMappings : اشاره گر به شیئی که حاوی آرایه ای از نوع ساختمان داده SHNAMEMAPPING است . هر یک از عضوهای این آرایه حاوی نام های قدیم و جدید فایلهایی است که کپی یا جابجا شده یا تغییر نام یافته اند. این آرایه تنها زمانی از اطلاعات مذکور پر می شود که علامت FOF_WANTMAPPINGHANDLE در پارامتر fFlags بکار رفته باشد. این اشاره گر باید بعد از استفاده توسط تابع SHFreeNameMappings آزاد شود.

❑ lpszProgressTitle : آدرس یک متغیر رشته ای که حاوی عنوان پنجره است. این پارامتر تنها زمانی مورد استفاده قرار می گیرد که پارامتر fFlags حاوی مقدار FOF_SIMPLEPROGRESS باشد.

☜ توجه : نگارش ۳۲ بیتی ویندوز، ساختمان های داده را در محدوده های دوکلمه ای (دو در دو بایت) قرار می دهد، اما بعضی از توابع API مانند SHFileOperation انتظار دارند داده های ورودی به این توابع بصورت تک بایتی باشند. اگر ساختمان داده SHFILEOPSTRUCT را بدون در نظر گرفتن این مساله به تابع فوق ارسال نمایید پارامترهای fAnyOperationsAborted ، hNameMappings و lpszProgressTitle از این ساختمان داده بدرستی ارسال نخواهند شد. اگر بخواهید پنجره کپی فایل عنوان دلخواه شما را که در

lpszProgressTitle ذخیره نموده اید نمایش دهد از یک آرایه از جنس بایت استفاده نمایید. روش استفاده از یک آرایه از جنس بایت در مثال زیر نشان داده شده است.

در بخش زیر خواهید دید که چگونه از تابع SHFileOperation برای کپی کردن فایل اجرایی اصلی ویژوال بیسیک و یک فایل Help به درون یک پوشه آزمایشی استفاده نمایید. به علاوه، این مثال نشان می دهد که چگونه یک پیغام دلخواه در پنجره نشان دهنده میزان پیشرفت عملیات نمایش دهید.

۱- در محیط ویژوال بیسیک یک پروژه جدید از نوع EXE بوجود آورید. Form1 بطور پیش فرض بوجود خواهد آمد.

۲- دو Check Box و یک Command Button به Form1 اضافه نمایید.

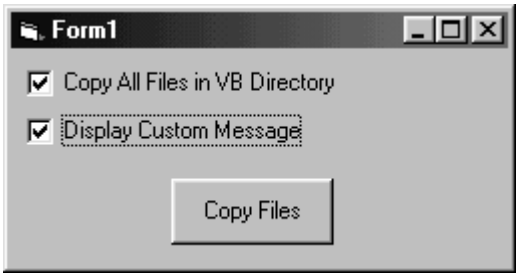
۳- لیست زیر را در Form1 وارد کنید:

```
Option Explicit
Private Const FO_COPY = &H2&   'Copies the files specified
                               'in the pFrom member to the
                               'location specified in the
                               'pTo member.

Private Const FO_DELETE = &H3& 'Deletes the files specified
                               'in pFrom (pTo is ignored.)

Private Const FO_MOVE = &H1&   'Moves the files specified
                               'in pFrom to the location
                               'specified in pTo.

Private Const FO_RENAME = &H4& 'Renames the files
                               'specified in pFrom.

Private Const FOF_ALLOWUNDO = &H40&   'Preserve Undo information.

Private Const FOF_CONFIRMMOUSE = &H2& 'Not currently implemented.

Private Const FOF_CREATEPROGRESSDLG = &H0& 'handle to the parent
                                           'window for the
                                           'progress dialog box.

Private Const FOF_FILESONLY = &H80&        'Perform the operation
                                           'on files only if a
                                           'wildcard file name
                                           '(*.*) is specified.

Private Const FOF_MULTIDESTFILES = &H1&    'The pTo member
                                           'specifies multiple
                                           'destination files (one
                                           'for each source file)
                                           'rather than one
```

```
                                        'directory where all
                                        'source files are
                                        'to be deposited.

Private Const FOF_NOCONFIRMATION = &H10&   'Respond with Yes to
                                        'All for any dialog box
                                        'that is displayed.

Private Const FOF_NOCONFIRMMKDIR = &H200&  'Does not confirm the
                                        'creation of a new
                                        'directory if the
                                        'operation requires one
                                        'to be created.

Private Const FOF_RENAMEONCOLLISION = &H8& 'Give the file being
                                        'operated on a new name
                                        'in a move, copy, or
                                        'rename operation if a
                                        'file with the target
                                        'name already exists.

Private Const FOF_SILENT = &H4&            'Does not display a
                                        'progress dialog box.

Private Const FOF_SIMPLEPROGRESS = &H100&  'Displays a progress
                                        'dialog box but does
                                        'not show the
                                        'file names.

Private Const FOF_WANTMAPPINGHANDLE = &H20&
                            'If FOF_RENAMEONCOLLISION is specified,
                            'the hNameMappings member will be filled
                            'in if any files were renamed.

' The SHFILOPSTRUCT is not double-word aligned. If no steps are
' taken, the last 3 variables will not be passed correctly. This
' has no impact unless the progress title needs to be changed.

Private Type SHFILEOPSTRUCT
    hwnd As Long
    wFunc As Long
    pFrom As String
    pTo As String
    fFlags As Integer
    fAnyOperationsAborted As Long
    hNameMappings As Long
    lpszProgressTitle As String
End Type

Private Declare Sub CopyMemory Lib "KERNEL32" Alias "RtlMoveMemory" _
               (hpvDest As Any, hpvSource As Any, ByVal cbCopy As Long)

Private Declare Function SHFileOperation Lib "Shell32.dll" _
               Alias "SHFileOperationA" (lpFileOp As Any) As Long

Private Sub Form_Load()
    Check1.Caption = "Copy All Files in VB Directory"
    Check2.Caption = "Display Custom Message"
    Command1.Caption = "Copy Files"
```

```
End Sub

Private Sub Command1_Click()
    Dim result As Long
    Dim lenFileop As Long
    Dim foBuf() As Byte
    Dim fileop As SHFILEOPSTRUCT

    lenFileop = LenB(fileop)    ' double word alignment increase
    ReDim foBuf(1 To lenFileop) ' the size of the structure.

    With fileop
        .hwnd = Me.hwnd

        .wFunc = FO_COPY

        ' The files to copy separated by Nulls and terminated by two
        ' nulls
        If Check1.Value = vbChecked Then
            .pFrom = "C:\Program Files\DevStudio\VB\*.*" & vbNullChar _
                     & vbNullChar
            .fFlags = FOF_SIMPLEPROGRESS Or FOF_FILESONLY
        Else
            .pFrom = "C:\PROGRAM FILES\DEVSTUDIO\VB\VB5.EXE" _
                     & vbNullChar _
                     & "C:\PROGRAM FILES\DEVSTUDIO\VB\README.HLP" _
                     & vbNullChar _
                     & vbNullChar
         End If

         .pTo = "C:\testfolder\" & vbNullChar & vbNullChar

         If Check2.Value = vbChecked Then
             .fFlags = FOF_SIMPLEPROGRESS Or FOF_NOCONFIRMATION Or _
                       FOF_NOCONFIRMMKDIR
             .lpszProgressTitle = "Your custom dialog string " & _
                                  "appears here." & vbNullChar _
                                                  & vbNullChar
          End If
    End With

    ' Now we need to copy the structure into a byte array
    Call CopyMemory(foBuf(1), fileop, lenFileop)

    ' Next we move the last 12 bytes by 2 to byte align the data
    Call CopyMemory(foBuf(19), foBuf(21), 12)
    result = SHFileOperation(foBuf(1))

    If result <> 0 Then  ' Operation failed
        MsgBox Err.LastDllError 'Show the error returned from
                                'the API.
    Else
          If fileop.fAnyOperationsAborted <> 0 Then
              MsgBox "Operation Failed"
          End If
      End If
End Sub
```

۴- از منوی Run گزینه Start را انتخاب کرده یا کلید F5 را فشار دهید تا برنامه اجرا شود. روی Copy Files کلیک کنید تا پروسه کپی آغاز شود. برای اینکه تمامی فایلهای موجود در دایرکتوری ویژوال بیسیک کپی شود، گزینه Copy All Files in VB Directory را کلیک کنید. برای اینکه ببینید پیغام دلخواه شما کجا نمایش داده می شود گزینه Display Custom Message را کلیک کنید.



شکل 3- نمای پنجره اصلی مثال در هنگام اجرا

☜ توجه : روی کامپیوترهای سریع ممکن است پنجره کپی نمایش داده نشود چرا که عملیات کپی پیش از نمایش داده شدن پنجره خاتمه می یابد. برای اینکه مطمئن باشید پنجره کپی نمایش داده می شود، گزینه Copy All Files in VB Directory را انتخاب نمایید.

پاک کردن فایل و قرار دادن آن در سطل بازیافت[۷] ویندوز ۹۸/۹۵

نام فایل	تاریخ	نگارش ویژوال بیسیک	نگارش ویندوز
154005.TXT	۲۴ آگوست ۱۹۹۸	4 , 5 , 6	95 / 98 / NT

در محیط ویندوز **۹۵** یا ویندوز **۹۸** این امکان وجود دارد که فایلها را پس از پاک کردن در سطل بازیافت ویندوز قرار دهیم تا کاربر بتواند در صورت نیاز فایل مورد نظر خود را بازیافت نماید. تابع Kill این امکان را در اختیار برنامه نویسان ویژوال بیسیک قرار نمی دهد، اما تابع SHFileOperation که از توابع API است این توانایی را دارد. مثالی که در زیر ارایه شده است نشان می دهد که چگونه از تابع فوق در ویژوال بیسیک استفاده می شود.

۱- در ویژوال بیسیک یک پروژه جدید ایجاد کنید. Form1 بطور پیش فرض بوجود می‌آید.

۲- در Form1 یک Command Button قرار دهید.

۳- لیست زیر را به Form1 اضافه نمایید:

```
Option Explicit

Private Type SHFILEOPSTRUCT
    hwnd As Long
    wFunc As Long
    pFrom As String
    pTo As String
    fFlags As Integer
    fAnyOperationsAborted As Boolean
    hNameMappings As Long
    lpszProgressTitle As String
End Type

Private Declare Function SHFileOperation Lib "shell32.dll" Alias _
        "SHFileOperationA" (lpFileOp As SHFILEOPSTRUCT) As Long
Private Declare Function GetTempFileName Lib "kernel32" _
        Alias "GetTempFileNameA" (ByVal lpszPath As String, _
        ByVal lpPrefixString As String, ByVal wUnique As Long, _
        ByVal lpTempFileName As String) As Long

Private Const FO_DELETE = &H3
Private Const FOF_ALLOWUNDO = &H40

Sub Command1_Click()
    Dim FileOperation As SHFILEOPSTRUCT
```

[۷] Recycle Bin

```
    Dim lReturn As Long
    Dim sTempFilename As String * 100
    Dim sSendMeToTheBin As String
    lReturn = GetTempFileName("c:\", "VB_", 0, sTempFilename)
    sSendMeToTheBin = Left(sTempFilename, InStr(sTempFilename, _
           Chr$(0)))
    With FileOperation
        .wFunc = FO_DELETE
        .pFrom = sSendMeToTheBin
        .fFlags = FOF_ALLOWUNDO
    End With
    lReturn = SHFileOperation(FileOperation)
    MsgBox "View your Recycle Bin for files beginning with VB_"
End Sub
```

۴- برای اجرای برنامه کلید F5 را فشار دهید و روی Command Button کلیک نمایید. در این زمان از شما در مورد پاک کردن فایل تایید گرفته می شود. در انتها در سطل بازیافت ویندوز شما فایلی با پسوند VB_ وجود خواهد داشت که توسط تابع GetTempFileName ایجاد شده است.

شکل 4 - پنجره تایید برای حذف فایل

استخراج اطلاعات مربوط به زمان آخرین دسترسی به فایل

نام فایل	تاریخ	نگارش ویژوال بیسیک	نگارش ویندوز
154821.TXT	۲۴ آگوست ۱۹۹۸	4 , 5 , 6	95 / 98 / NT

ویندوزهای ۹۵/۹۸/NT اطلاعات زیادی در مورد یک فایل ذخیره می کنند، از جمله تاریخ و ساعت آخرین دسترسی به هر فایل. این اطلاعات از طریق تابع FileDateTime قابل دسترس نیستند ولی به کمک ترکیبی از توابع API قابل استخراج خواهند بود. در این بخش روش انجام این کار را خواهیم دید.

زمان ایجاد، زمان آخرین دفعه نوشته شدن در فایل و زمان آخرین دسترسی به فایل (خواندن) همگی در سیستم ویندوز ۳۲ بیتی ذخیره می شوند، اما فقط تاریخ و زمان آخرین تغییرات در فایل مستقیما از درون ویژوال بیسیک قابل دسترسی هستند. تابع API مورد استفاده انتظار دارد به جای شماره فایل که در ویژوال بیسیک استفاده می شود، یک شناسه فایل از نوع شناسه هایی که سیستم عامل بکار می برد، دریافت نماید. اما به دلیل محدودیتی که در ویژوال بیسیک ۳۲ بیتی وجود دارد، تابع FileAttr قادر نیست شناسه فایل سیستم عامل را برگرداند و به همین دلیل برای گشودن فایل لازم است از توابع API استفاده شود. برای به دست آوردن این اطلاعات مطابق مثال زیر عمل نمایید:

۱- در ویژوال بیسیک یک پروژه جدید ایجاد کنید. Form1 بطور پیش فرض بوجود می‌آید.

۲- به Form1 یک Text Box اضافه نمایید. نام پیش فرض آن Text1 خواهد بود.

۳- یک Command Button به Form1 اضافه نمایید. نام پیش فرض آن Command1 است.

۴- لیست زیر را به بخش declarations از Form1 اضافه نمایید:

```
Option Explicit

Private Const OF_READ = &H0
Private Const OF_SHARE_DENY_NONE = &H40
Private Const OFS_MAXPATHNAME = 128

Private Type OFSTRUCTREC
   cBytes As Byte
   fFixedDisk As Byte
   nErrCode As Integer
```

```
   Reserved1 As Integer
   Reserved2 As Integer
   szPathName(OFS_MAXPATHNAME) As Byte
End Type

Private Type FILETIMEREC
   dwLowDateTime As Long
   dwHighDateTime As Long
End Type

Private Type SYSTEMTIMEREC
   wYear As Integer
   wMonth As Integer
   wDayOfWeek As Integer
   wDay As Integer
   wHour As Integer
   wMinute As Integer
   wSecond As Integer
   wMilliseconds As Integer
End Type

Private Declare Function FileTimeToSystemTime Lib "kernel32" _
     (lpFileTime As FILETIMEREC, lpSystemTime As SYSTEMTIMEREC) As Long
Private Declare Function GetFileTime Lib "kernel32" (ByVal _
     hFile As Long, lpCreationTime As FILETIMEREC, lpLastAccessTime _
     As FILETIMEREC, lpLastWriteTime As FILETIMEREC) As Long
Private Declare Function OpenFile Lib "kernel32" (ByVal lpFileName As _
     String, lpReOpenBuff As OFSTRUCTREC, ByVal wStyle As Long) As Long
Private Declare Function hread Lib "kernel32" Alias "_hread" _
     (ByVal hFile As Long, lpBuffer As Any, ByVal lBytes As Long) As Long
Private Declare Function lclose Lib "kernel32" Alias "_lclose" (ByVal _
     hFile As Long) As Long

Sub Form_Load()
     Command1.Caption = "&Get file access time"
     Text1.Text = "C:\AUTOEXEC.BAT"
End Sub

Private Sub Command1_Click()
   Dim sInpFile As String
   Dim hFile As Integer
   Dim FileStruct As OFSTRUCTREC
   Dim iRC As Integer
   Dim CreationTime As FILETIMEREC
   Dim LastAccessTime As FILETIMEREC
   Dim LastWriteTime As FILETIMEREC
   Dim SystemTime As SYSTEMTIMEREC
   sInpFile = Trim(Text1.Text)
   ' check that the file exists
   If Len(Dir(sInpFile)) = 0 Then
      MsgBox "Can't find the file", vbExclamation
      Exit Sub
   End If

   ' Open it to get a stream handle
    hFile = OpenFile(sInpFile, FileStruct, OF_READ Or OF_SHARE_DENY_NONE)
    If hFile = 0 Then
       MsgBox "Can't open the file", vbExclamation
       Exit Sub
```

```
    End If

    If GetFileTime(hFile, CreationTime, _
       LastAccessTime, LastWriteTime) Then
                ' massage time into format that we can use
       If Not FileTimeToSystemTime(LastAccessTime, SystemTime) Then
          Print "Year of file  :" & SystemTime.wYear
          Print "Month of File :" & SystemTime.wMonth
          Print "Day of File   :" & SystemTime.wDay
       Else
          MsgBox "FileTimeToSystemTime Failed"
       End If
    Else
       MsgBox "GetFileTime Failed"
    End If

    iRC = lclose(hFile)
End Sub
```

۵- برای اجرای برنامه F5 را فشار دهید یا روی دکمه مربوطه کلیک نمایید. در Text Box یک نام فایل وارد نمایید، در صورت نیاز مسیر فایل را نیز وارد نمایید و در نهایت روی Command Button کلیک نمایید.

توجه داشته باشید که آخرین زمان تغییرات در فایل و همچنین زمان ایجاد فایل بازگردانده می شود. می‌توان با ایجاد تغییراتی در لیست برنامه فوق این امکان را فراهم نمود که هم زمان و هم تاریخ را مشخص نماید. این مقادیر در رکورد SYSTEMTIME بازگردانده می شوند.

سیستم های نگهداری فایل FAT و New Technology امکان ذخیره زمان ایجاد، زمان آخرین تغییرات و زمان آخرین دسترسی به فایل را دارند. در محیط ویندوز ۹۵، دقت زمانی برای فایلی در سیستم FAT به میزان ۲ ثانیه است. دقت زمانی برای فایلهایی که در سیستم های دیگری مانند شبکه ها ذخیره می شوند بستگی به سیستم فایل مربوطه دارد ولی ممکنست محدودیت های سیستم عامل کامپیوترهای متصل به شبکه نیز در این مسئله موثر باشند.

تبدیل یک نام فایل کوتاه به معادل طولانی آن

نام فایل	تاریخ	نگارش ویژوال بیسیک	نگارش ویندوز
154822.TXT	۷ آگوست ۱۹۹۸	4 , 5 , 6	95 / 98 / NT

می توان از تابع ( )Dir برای دریافت یک نام فایل طولانی استفاده نمود ولی در این صورت این نام فایل شامل مسیر محل قرار گیری فایل نخواهد بود. با شکستن یک نام فایل با فرمت کوتاه به دایرکتوریهای تشکیل دهنده آن می توانید برای ایجاد یک نام فایل با فرمت بلند از تابع مذکور استفاده نمایید. این بخش نشان می دهد چگونه این کار قابل انجام است.

مثال زیر حاوی تابعی است که نام فایل با فرمت کوتاه را به معادل بلند آن تبدیل می کند و ضمنا مثالی از کاربرد این تابع را نیز نشان می دهد.

۱- ویژوال بیسیک را اجرا نمایید. Form1 به صورت پیش فرض بوجود می آید.

۲- روی Form1 یک Command Button قرار دهید.

۳- از منوی Insert گزینه Module را انتخاب کنید تا یک ماژول به پروژه اضافه شود.

۴- لیست زیر را در Module1 وارد نمایید:

```
Public Function GetLongFilename (ByVal sShortName As String) As String
   Dim sLongName As String
   Dim sTemp As String
   Dim iSlashPos As Integer

   'Add \ to short name to prevent Instr from failing
   sShortName = sShortName & "\"

   'Start from 4 to ignore the "[Drive Letter]:\" characters
   iSlashPos = InStr(4, sShortName, "\")
   'Pull out each string between \ character for conversion
   While iSlashPos
       sTemp = Dir(Left$(sShortName, iSlashPos - 1), _
          vbNormal + vbHidden +    vbSystem + vbDirectory)
       If sTemp = "" Then
          'Error 52 - Bad File Name or Number
          GetLongFilename = ""
          Exit Function
       End If
       sLongName = sLongName & "\" & sTemp
       iSlashPos = InStr(iSlashPos + 1, sShortName, "\")
   Wend

   'Prefix with the drive letter
```

```
   GetLongFilename = Left$(sShortName, 2) & sLongName

End Function
```

۵- لیست زیر را به واقعه Command1_Click اضافه نمایید:

```
Private Sub Command1_Click()
   'Assumes C:\Program Files\Common Files is a valid path
   Print GetLongFilename("C:\PROGRA~1\COMMON~1")
End Sub
```

۶- برای اجرای برنامه کلید F5 را فشار دهید یا از منوی Run گزینه Start را انتخاب کنید.

۷- روی Command Button کلیک کنید.

در صورتیکه مسیر فایلی که مشخص کرده اید درست باشد، معادل بلند آن روی فرم نمایش داده خواهد شد. در صورتیکه مسیر فایل درست نباشد، چیزی نمایش داده نمی شود، که در این صورت لازمست یک نام فایل کوتاه درست در لیست مربوط به واقعه Command1_Click وارد نمایید.

استفاده از پنجره دیالوگ بازیابی فایل توسط API

نام فایل	تاریخ	نگارش ویژوال بیسیک	نگارش ویندوز
161286.TXT	۷ آگوست ۱۹۹۸	4 , 5 , 6	95 / 98 / NT

در این بخش با ارائه یک مثال خواهید دید که چگونه از پنجره دیالوگ بازیابی فایل که در Comdlg32.dll قرار دارد، استفاده نمایید. استفاده از Comdlg32.OCX روش خوبی برای استفاده از امکانات ویندوز است، زیرا ویندوز همیشه فایل مرتبط به آن یعنی Comdlg32.dll را در حافظه آماده دارد. استفاده از این روش آسان است ولی سرعت اجرای برنامه را کاهش می دهد. اگر از ابزار نوع OCX استفاده نمایید، مجبور خواهید بود ماژول مربوطه را به درون حافظه بار کنید و به علاوه، یک فایل ۹۰ کیلوبایتی OCX را در اختیار استفاده کنندگان از نرم افزار خود

قرار دهید. برای بهبود بخشیدن به بازده و سرعت، باید میزان استفاده از ابزارها را در برنامه خود به حداقل برسانید. در عوض، می توانید بطور مستقیم از توابع API استفاده نمایید.

در صورتیکه از فراخوانی توابع API استفاده نمایید، بعضی از توانایی هایی که Comdlg32.ocx در اختیار می گذارد، مانند دکمه Help ، را از دست خواهید داد. اگر به دکمه Help نیاز دارید، لازمست از Comdlg32.OCX استفاده کنید.

۱- ویژوال بیسیک را اجرا کنید و یک پروژه جدید ایجاد کنید. به این ترتیب Form1 بطور پیش فرض ایجاد خواهد شد.

۲- روی Form1 یک Command Button قرار دهید.

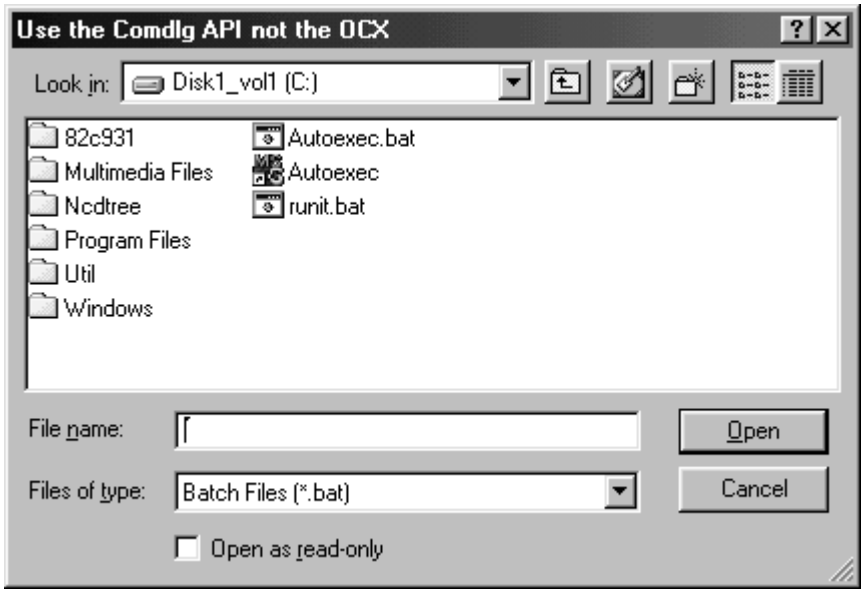
۳- لیست زیر را در Form1 وارد نمایید:

```
Option Explicit

Private Declare Function GetOpenFileName Lib "comdlg32.dll" Alias _
         "GetOpenFileNameA" (pOpenfilename As OPENFILENAME) As Long

Private Type OPENFILENAME
    lStructSize As Long
    hwndOwner As Long
    hInstance As Long
    lpstrFilter As String
    lpstrCustomFilter As String
    nMaxCustFilter As Long
    nFilterIndex As Long
    lpstrFile As String
    nMaxFile As Long
    lpstrFileTitle As String
    nMaxFileTitle As Long
    lpstrInitialDir As String
    lpstrTitle As String
    flags As Long
    nFileOffset As Integer
    nFileExtension As Integer
    lpstrDefExt As String
    lCustData As Long
    lpfnHook As Long
    lpTemplateName As String
End Type

Private Sub Command1_Click()
    Dim OpenFile As OPENFILENAME
    Dim lReturn As Long
    Dim sFilter As String
    OpenFile.lStructSize = Len(OpenFile)
    OpenFile.hwndOwner = Form1.hWnd
    OpenFile.hInstance = App.hInstance
    sFilter = "Batch Files (*.bat)" & Chr(0) & "*.BAT" & Chr(0)
    OpenFile.lpstrFilter = sFilter
```

```
    OpenFile.nFilterIndex = 1
    OpenFile.lpstrFile = String(257, 0)
    OpenFile.nMaxFile = Len(OpenFile.lpstrFile) - 1
    OpenFile.lpstrFileTitle = OpenFile.lpstrFile
    OpenFile.nMaxFileTitle = OpenFile.nMaxFile
    OpenFile.lpstrInitialDir = "C:\"
    OpenFile.lpstrTitle = "Use the Comdlg API not the OCX"
    OpenFile.flags = 0
    lReturn = GetOpenFileName(OpenFile)
    If lReturn = 0 Then
        MsgBox "The User pressed the Cancel Button"
    Else
        MsgBox "The user Chose " & Trim(OpenFile.lpstrFile)
    End If
End Sub
```

۴- برای اجرای برنامه کلید F5 را فشار دهید و روی Command Button کلیک نمایید. در این مرحله پنجره دیالوگ بازیابی فایل ظاهر خواهد شد.

Win32 SDK مستندات بیشتری در زمینه فراخوانی توابع دیگر موجود در Comdlg32.dll مانند ChooseFont، ChooseColor و GetSaveFileName را در اختیار می گذارد.



شکل 5 - نمونه ای از پنجره دیالوگ فایل ایجاد شده توسط API