

```

1  ' PicInfo bitmap file information class
2  ' ©2006 Aivosto Oy (www.aivosto.com)
3  '
4  ' This file is part of a sample project for Project Analyzer.
5  ' Distribution of this file is only allowed along with Project Analyzer
6  ' according to the Project Analyzer license terms.
7
8  Option Explicit
9
10 Implements IPicInfo
11
12 Private Enum EBMPType
13     bmpUnknown
14     bmpInfoHeader
15     bmpCoreHeader
16 End Enum
17 Private BMPType As EBMPType
18
19 Private Type BITMAPFILEHEADER
20     bfType As Integer ' Specifies the file type, must be BM
21     bfSize As Long ' Specifies the size, in bytes, of the bitmap file.
22     bfReserved1 As Integer ' Reserved; must be zero.
23     bfReserved2 As Integer ' Reserved; must be zero.
24     bfoffBits As Long ' Specifies the offset, in bytes, from the beginning of the
        BITMAPFILEHEADER structure to the bitmap bits.
25 End Type
26 Private Type BITMAPINFOHEADER
27     biSize As Long
28     biWidth As Long
29     biHeight As Long
30     biPlanes As Integer
31     biBitCount As Integer
32     biCompression As Long
33     biSizeImage As Long
34     biXPelsPerMeter As Long
35     biYPelsPerMeter As Long
36     biClrUsed As Long
37     biClrImportant As Long
38 End Type
39 Private Type BITMAPCOREHEADER
40     bcSize As Long
41     bcWidth As Integer
42     bcHeight As Integer
43     bcPlanes As Integer
44     bcBitCount As Integer
45 End Type
46 Private InfoHeader As BITMAPINFOHEADER
47 Private CoreHeader As BITMAPCOREHEADER
48
49 Private Const BI_RGB = 0&
50 Private Const BI_RLE4 = 2&
51 Private Const BI_RLE8 = 1&
52 Private Const BI_bitfields = 3&
53
54 ' IsRLE indicates whether the bitmap is RLE compressed (True) or not (False)
55 ' This variable is written but not read
56 ' It might well be removed without affecting the functionality of the program
57 Private IsRLE As Boolean
58
59 ' Filename used in the previous call to ReadFile
60 Private StoredFilename As String
61
62
63 Private Function ReadBitmapFile(ByVal Filename As String) As Boolean
64 ' Read picture information from a bitmap file of type BMP/DIB/RLE
65 ' [Filename] Name of picture file
66 ' Return value:
67 ' True - Picture information retrieved
68 ' False - Not a valid bitmap file
69
70 Dim BMPFileHeader As BITMAPFILEHEADER ' Main BM file header
71 Dim FileNr As Integer ' Number of open file
72 Dim HeaderSize As Long ' Size of header data in bytes
73
74 IsRLE = False ' Clear the IsRLE flag. We will set it below to True if required.
75 BMPType = bmpUnknown ' Set file type to bmpUnknown until we can verify the real type
76
77 ' Open the file for binary read access
78 FileNr = FreeFile
79 Open Filename For Binary Access Read Lock Write As #FileNr
80
81 If LOF(FileNr) > Len(BMPFileHeader) Then
82     ' File length is "enough", read main BM file header header
83     Get #FileNr, 1, BMPFileHeader
84     If BMPFileHeader.bfType = &H4D42 Then ' BM
85         If BMPFileHeader.bfReserved1 = 0 And BMPFileHeader.bfReserved2 = 0 Then
86             ' Signature OK
87
88             ' Retrieve size of following header
89             Get #FileNr, , HeaderSize
90             Seek #FileNr, Seek(FileNr) - 4 ' Rewind to start of header
91
92             Select Case HeaderSize
93                 Case Len(CoreHeader)
94                     Get #FileNr, , CoreHeader
95                     BMPType = bmpCoreHeader
96                     ReadBitmapFile = True ' File valid
97                 Case Len(InfoHeader)
98                     Get #FileNr, , InfoHeader
99                     BMPType = bmpInfoHeader
100
101                 ' Determine bitmap compression
102                 Select Case InfoHeader.biCompression
103                     Case BI_RLE8, BI_RLE4
104                         ' The bitmap is RLE compressed
105                         IsRLE = True
106                 End Select
107                 ReadBitmapFile = True ' File valid

```

```
108     Case Else
109         ReadBitmapFile = False ' Invalid/unsupported file type
110     End Select
111 End If
112 End If
113 End If
114 Close FileNr
115
116 End Function
117
118
119 Private Property Get IPicInfo_FileName() As String
120     ' Returns the filename used in the previous call to ReadFile
121
122     IPicInfo_FileName = StoredFilename
123
124 End Property
125
126 Private Property Get IPicInfo_MaxColors() As Variant
127
128     Select Case BMPType
129     Case bmpCoreHeader
130         IPicInfo_MaxColors = 2 ^ CoreHeader.bcBitCount
131     Case bmpInfoHeader
132         IPicInfo_MaxColors = 2 ^ InfoHeader.biBitCount
133     End Select
134
135
136 End Property
137
138 Private Function IPicInfo_ReadFile(ByVal Filename As String) As Boolean
139     ' Read a picture file to retrieve picture information
140     ' [Filename] File to read
141     ' Return value:
142     ' True - Picture information retrieved
143     ' False - Error, information not retrieved
144
145     IPicInfo_ReadFile = ReadBitmapFile(Filename)
146
147 End Function
148
149
150 Private Property Get IPicInfo_Size() As TPicSize
151     ' Returns picture width and height in pixels
152
153     Select Case BMPType
154     Case bmpCoreHeader
155         IPicInfo_Size.Width = CoreHeader.bcWidth
156         IPicInfo_Size.Height = CoreHeader.bcHeight
157     Case bmpInfoHeader
158         IPicInfo_Size.Width = InfoHeader.biWidth
159         IPicInfo_Size.Height = InfoHeader.biHeight
160     End Select
161
```

```
162 End Property
163
164
```

```

1  ' PicInfo main form
2  ' ©2006 Aivosto Oy (www.aivosto.com)
3  '
4  ' This file is part of a sample project for Project Analyzer.
5  ' Distribution of this file is only allowed along with Project Analyzer
6  ' according to the Project Analyzer license terms.
7
8  Option Explicit
9
10 Private Sub DisplayPictureInfo(ByVal Filename As String)
11 ' Display information about a picture file
12 ' [Filename] Name of picture file
13
14 Dim IPicInfo As IPicInfo ' Object that implements the IPicInfo interface
15 Dim PicSize As TPicSize ' Picture size in pixels
16 Dim MaxColors As Variant ' Maximum number of colors in this picture
17
18 ' Determine file type
19 Filename = LCase$(Filename)
20 If Filename Like "*.gif" Then
21 ' GIF file, get a PicGIF object to handle it
22 Set IPicInfo = New PicGIF
23 Else
24 ' Assume it is a bitmap file, get a PicBMP object to handle it
25 Set IPicInfo = New PicBMP
26 End If
27
28 ' Read the picture file
29 If IPicInfo.ReadFile(Filename) Then
30 ' Retrieve picture size in pixels
31 PicSize = IPicInfo.Size
32 ' Retrieve maximum number of colors in this picture
33 MaxColors = IPicInfo.MaxColors
34
35 ' Display retrieved information
36 Me.FileName = "File: " & Filename
37 Me.Size = "Size: " & PicSize.Width & " x " & PicSize.Height & " pixels"
38 Me.MaxColors = "Max colors: " & MaxColors
39
40 ' Display the picture as well
41 Pict.Picture = LoadPicture(Filename)
42 End If
43
44 End Sub
45
46 Private Sub Form_KeyPress(KeyAscii As Integer)
47
48 If KeyAscii = 15 Then
49 ' Ctrl+O = Open
50 OpenFile_Click
51 End If
52
53 End Sub
54

```

```

55 Private Sub Form_Load()
56 ' As the form loads, _
57 show the program title in the form caption
58
59 Me.Caption = ProgramTitle ' Global variable access
60
61 End Sub
62
63
64 Private Sub OpenFile_Click()
65 ' User has clicked the "Open picture..." command button
66
67 Dim Filename As String
68
69 ' Ask the user to select a picture file
70 Filename = ShowFileOpenDialog(Me.hWnd, "", "Picture files (*.gif;*.bmp)|*.gif;*.bmp")
71
72 If Len(Filename) Then
73 ' We have received a filename to open
74 DisplayPictureInfo Filename
75 End If
76
77 End Sub
78
79
80 Private Sub Pict_OLEDragDrop(Data As DataObject, Effect As Long, Button As Integer, Shift As
Integer, X As Single, Y As Single)
81 ' If user has dropped one or more files on the picture area
82 ' try to load the first file and show its picture info
83
84 If Data.GetFormat(vbCFFiles) Then
85 Effect = vbDropEffectCopy
86 DisplayPictureInfo Data.Files(1)
87 End If
88
89 End Sub
90
91 Private Sub Pict_OLEDragOver(Data As DataObject, Effect As Long, Button As Integer, Shift As
Integer, X As Single, Y As Single, State As Integer)
92 ' If user has dragged one or more files, indicate we will accept them
93
94 If Data.GetFormat(vbCFFiles) Then
95 Effect = vbDropEffectCopy
96 End If
97
98 End Sub
99
100
101 Private Sub SaveFile_Click()
102 ' Save file not implemented
103
104 Stop
105
106 End Sub

```

107

```

1  ' PicInfo GIF file information class
2  ' ©2006 Aivosto Oy (www.aivosto.com)
3  '
4  ' This file is part of a sample project for Project Analyzer.
5  ' Distribution of this file is only allowed along with Project Analyzer
6  ' according to the Project Analyzer license terms.
7
8  Option Explicit
9
10 Implements IPicInfo
11
12 Private StoredFilename As String ' Filename used in the previous call to ReadFile
13
14 ' FileOK is a flag indicating whether a valid GIF file has been read
15 ' True - The variables below contain valid data
16 ' False - The variables below contain undefined data and should not be used
17 Private FileOK As Boolean
18
19 ' Brief introduction to GIF file structure
20 '
21 ' Every GIF file starts with the following bytes:
22 ' 6 bytes for Signature (either GIF87a or GIF89a in ASCII)
23 ' 7 bytes for LogicalScreenDescriptor
24
25 Private Signature As String * 6 ' GIF87a, GIF89a
26
27 Private Type TLogicalScreenDescriptor
28     Width As Integer ' uint - Width, in pixels
29     Height As Integer ' uint - Height, in pixels
30     Packed As Byte ' Byte packed with color information
31     BackgroundColorIndex As Byte ' Background Color index to Global Color Table
32     PixelAspectRatio As Byte ' Approximation of aspect ratio of pixel in the original image
33 End Type
34 Private LogicalScreenDescriptor As TLogicalScreenDescriptor ' Storage for LogicalScreenDescriptor
35 Private MaxColors As Long ' Maximum number of colors in .gif
36
37 Private Function UInt(ByVal i As Integer) As Long
38 ' Convert integer to unsigned integer
39 ' [i] Signed integer to convert to an unsigned value
40 ' Return value:
41 ' Unsigned integer (fits in the Long datatype)
42
43 If i < 0 Then
44     UInt = 65536 + i
45 Else
46     UInt = i
47 End If
48
49 End Function
50
51
52
53 Private Function ReadGIF(ByVal GIFFilename As String) As Boolean
54 ' Reads a GIF file and retrieves image information
55 ' [GIFFilename] Name of file to read
56 ' Return value:
57 ' True - GIF file was read and information retrieved
58 ' False - Error or invalid file
59
60 Dim ColorResolution As Integer ' Number of color bits in the file (1..8)
61 Dim FileNr As Integer ' Number of open data file
62
63 StoredFilename = GIFFilename ' Store Filename for later retrieval
64 FileOK = False ' Clear the FileOK flag until we have successfully read a valid .gif
65     file
66 ' Open the .gif file for binary read access
67 FileNr = FreeFile
68 Open GIFFilename For Binary Access Read Lock Write As #FileNr
69
70 If LOF(FileNr) > Len(Signature) + Len(LogicalScreenDescriptor) Then
71     ' The file size is "enough"
72
73     ' Read 6 GIF signature bytes, which must be either GIF87a or GIF89a in ASCII
74     Get #FileNr, 1, Signature
75     If Signature = "GIF87a" Or Signature = "GIF89a" Then
76         ' Signature bytes OK
77
78         ' Read logical screen descriptor, which contains picture size and color info
79         Get #FileNr, , LogicalScreenDescriptor
80
81         ' Determine max number of colors
82         ColorResolution = (LogicalScreenDescriptor.Packed And (64 + 32 + 16)) \ 16 + 1
83         MaxColors = 2 ^ ColorResolution
84
85         FileOK = True
86         ReadGIF = True
87     End If
88 End If
89 Close FileNr
90 FileNr = 0
91
92 End Function
93
94
95
96 Friend Property Get Version() As String
97 ' Returns GIF format version (either "GIF87a" or "GIF89a")
98
99 ' This procedure is not used by the sample project
100 ' Project Analyzer reports it as a dead procedure
101
102 If FileOK Then
103     Version = Signature
104 End If
105

```

```
106 End Property
107
108
109 Private Property Get IPicInfo_Filename() As String
110 ' Returns the filename used in the previous call to ReadFile
111
112 IPicInfo_Filename = StoredFilename
113
114 End Property
115
116 Private Property Get IPicInfo_MaxColors() As Variant
117 ' Returns the maximum number of colors possible to represent in picture
118
119 If FileOK Then
120     ' We have the color information available
121     IPicInfo_MaxColors = MaxColors
122 End If
123
124 End Property
125
126 Private Function IPicInfo_ReadFile(ByVal Filename As String) As Boolean
127 ' Read a picture file to retrieve picture information
128 ' [Filename] File to read
129 ' Return value:
130 ' True - Picture information retrieved
131 ' False - Error, information not retrieved
132
133 IPicInfo_ReadFile = ReadGIF(Filename)
134
135 End Function
136
137
138 Private Property Get IPicInfo_Size() As TPicSize
139 ' Returns picture width and height in pixels
140
141 If FileOK Then
142     ' We have retrieved the picture size
143
144     ' As the width and height are unsigned integers but VB6 supports no unsigned integer datatype,
145     ' we need to convert the values with UInt() before returning to caller.
146     IPicInfo_Size.Width = UInt(LogicalScreenDescriptor.Width)
147     IPicInfo_Size.Height = UInt(LogicalScreenDescriptor.Height)
148 End If
149
150 End Property
151
152
```

```
1 ' PicInfo IPicInfo interface
2 ' ©2006 Aivosto Oy (www.aivosto.com)
3 '
4 ' This file is part of a sample project for Project Analyzer.
5 ' Distribution of this file is only allowed along with Project Analyzer
6 ' according to the Project Analyzer license terms.
7
8 Option Explicit
9
10
11 ' Type for picture size
12 Public Type TPicSize
13     Width As Long ' Picture width in pixels
14     Height As Long ' Picture height in pixels
15 End Type
16
17 Public Property Get Filename() As String
18 Attribute Filename.VB_Description = "Returns the filename used in the previous call to ReadFile"
19 ' Returns the filename used in the previous call to ReadFile
20
21 End Property
22
23
24 Public Property Get MaxColors() As Variant
25 Attribute MaxColors.VB_Description = "Returns the maximum number of colors possible to represent in
26 picture"
27 ' Returns the maximum number of colors possible to represent in picture.
28 ' Return value:
29 ' The return value is always numeric. The maximum value may exceed
30 ' the size of the Long data type, therefore we use the Variant data type.
31 End Property
32
33 Public Function ReadFile(ByVal Filename As String) As Boolean
34 Attribute ReadFile.VB_Description = "Read a picture file to retrieve picture information"
35 ' Read a picture file to retrieve picture information
36 ' [Filename] File to read
37 ' Return value:
38 ' True - Picture information retrieved
39 ' False - Error, information not retrieved
40
41 End Function
42
43 Public Property Get Size() As TPicSize
44 Attribute Size.VB_Description = "Returns picture width and height in pixels"
45 ' Returns picture width and height in pixels
46
47 End Property
48
49
```

```

1  ' PicInfo main module
2  ' ©2006 Aivosto Oy (www.aivosto.com)
3  '
4  ' This file is part of a sample project for Project Analyzer.
5  ' Distribution of this file is only allowed along with Project Analyzer
6  ' according to the Project Analyzer license terms.
7
8  Option Explicit
9
10 ' Global variable containing the title of this program
11 Public ProgramTitle As String
12
13
14 ' API declarations for FileDialog
15 Private Declare Function GetOpenFileName Lib "comdlg32.dll" (pOpenFilename As OpenFilename) As Long
16
17 ' This API Declare is dead
18 Private Declare Function GetSaveFileName Lib "comdlg32.dll" (pOpenFilename As OpenFilename) As Long
19
20 Private Type OpenFilename
21     lStructSize As Long
22     hwndOwner As Long
23     hInstance As Long
24     lpstrFilter As String
25     lpstrCustomFilter As String
26     nMaxCustFilter As Long
27     nFilterIndex As Long
28     lpstrFile As String
29     nMaxFile As Long
30     lpstrFileTitle As String
31     nMaxFileTitle As Long
32     lpstrInitialDir As String
33     lpstrTitle As String
34     Flags As Long
35     nFileOffset As Integer
36     nFileExtension As Integer
37     lpstrDefExt As String
38     lCustData As Long
39     lpfnHook As Long
40     lpTemplateName As String
41 End Type
42
43 Public Enum EFileDialogFlags
44     OFN_ALLOWMULTISELECT = &H200 ' box allows multiple selections
45     OFN_CREATEPROMPT = &H2000 ' If the user specifies a file that does not exist, this
46     ' flag causes the dialog box to prompt the user for permission to create the file.
47     OFN_FILEMUSTEXIST = &H1000 ' user can type only names of existing files in the File
48     ' Name entry field
49     OFN_HIDEREADONLY = &H4 ' Hides the Read Only check box.
50
51     OFN_NOCHANGEDIR = &H8
52     ' Restores the current directory to its original value if the user changed the directory while
53     ' searching for files.
54     ' Windows NT 4.0/2000/XP: This flag is ineffective for GetOpenFileName.
55
56     OFN_NODEREFERENCELINKS = &H100000
57     ' Directs the dialog box to return the path and file name of the selected shortcut (.LNK) file.
58
59     OFN_NONETWORKBUTTON = &H20000
60     ' Hides and disables the Network button.
61
62     OFN_NOREADONLYRETURN = &H8000&
63     ' Specifies that the returned file does not have the Read Only check box selected and is not in
64     ' a write-protected directory.
65
66     OFN_NOTESTFILECREATE = &H10000
67     ' Specifies that the file is not created before the dialog box is closed.
68     ' This flag should be specified if the application saves the file on a create-nonmodify network
69     ' share.
70     ' When an application specifies this flag, the library does not check for write protection, a
71     ' full disk,
72     ' an open drive door, or network protection.
73
74     OFN_OVERWRITEPROMPT = &H2
75     ' Causes the Save As dialog box to generate a message box if the selected file already exists.
76     ' The user must confirm whether to overwrite the file.
77
78     OFN_PATHMUSTEXIST = &H800
79     ' Specifies that the user can type only valid paths and file names.
80
81     OFN_READONLY = &H1
82     ' Causes the Read Only check box to be selected initially when the dialog box is created.
83     ' This flag indicates the state of the Read Only check box when the dialog box is closed.
84
85     OFN_SHAREAWARE = &H4000
86     ' Specifies that if a call to the OpenFile function fails because of a network sharing
87     ' violation, the error is ignored and the dialog box returns the selected file name.
88
89     ' Open dialog defaults:
90     ' File and path must exist
91     ' No Read Only checkbox is displayed
92     OFN_OPENDEFAULTS = OFN_FILEMUSTEXIST Or OFN_PATHMUSTEXIST Or OFN_HIDEREADONLY
93
94     ' Save dialog defaults:
95     ' User must confirm overwrite
96     ' Path must exist
97     ' * Read-only files can be selected, caller must use error handling if need to
98     ' kill/overwrite/append
99     OFN_SAVEDEFAULTS = OFN_OVERWRITEPROMPT Or OFN_PATHMUSTEXIST Or OFN_HIDEREADONLY Or OFN_
100     ' NOREADONLYRETURN
101
102 End Enum
103
104 Const MAXFILE = 512 ' Size of file name buffer in chars. Should be at least 256.
105
106 ' Error code retrieval

```



```

99 Private Declare Function ComDlgExtendedError Lib "comdlg32.dll" () As Long
100 Private Const CDERR_DIALOGFAILURE = &HFFFF
101 Private Const CDERR_FINDRESFAILURE = &H6
102 Private Const CDERR_NOINSTANCE = &H4
103 Private Const CDERR_INITIALIZATION = &H2
104 Private Const CDERR_NOHOOK = &HB
105 Private Const CDERR_LOCKRESFAILURE = &H8
106 Private Const CDERR_NOTEMPLATE = &H3
107 Private Const CDERR_LOADRESFAILURE = &H7
108 Private Const CDERR_STRUCTSIZE = &H1
109 Private Const CDERR_LOADSTRFAILURE = &H5
110 Private Const FNERR_BUFFERTOOSMALL = &H3003
111 Private Const CDERR_MEMALLOCFailure = &H9
112 Private Const FNERR_INVALIDFILENAME = &H3002
113 Private Const CDERR_MEMLOCKFAILURE = &HA
114 Private Const FNERR_SUBCLASSFAILURE = &H3001
115
116
117 Public Function ShowFileOpenDialog(ByVal hwndOwner As Long, ByVal DefaultExtension As String,
    ByVal Filter As String, Optional ByRef FilterIndex As Long, Optional ByVal InitialDir As String,
    Optional ByVal DialogTitle As String, Optional ByRef Flags As EFileDialogFlags = OFN_OPENDEFAULTS) As
    String
118 ' Show the Open file dialog box using comdlg32.dll
119 ' [hwndOwner] In. Window handle of owner window. Can be zero.
120 ' [DefaultExtension] In. Default file extension such as "txt".
121 ' [Filter] In. Filters in format "File type/*.ext;*.txt|All files/*.*".
122 ' [FilterIndex] In/out. Selected file type in filter. First filter is number 1.
123 ' [InitialDir] In. Initial directory. If not given or empty, uses system default initial directory.
    See FileDialog for explanation.
124 ' [DialogTitle] In. Dialog caption. If not given or empty, uses system default caption.
125 ' [Flags] In/out. Flags for the dialog. Should be given as "OFN_OPENDEFAULTS or ...". On return
    contains any altered flags.
126 ' Return value:
127 ' Selected file name if user pressed OK
128 ' Empty string if user pressed Cancel or there was an error
129 ' Caller can verify error status by calling ComDlgExtendedError
130
131 ShowFileOpenDialog = FileDialog(hwndOwner, DefaultExtension, Filter, FilterIndex, InitialDir,
    DialogTitle, Flags)
132
133 End Function
134
135 Private Function FileDialog(ByVal hwndOwner As Long, ByVal DefaultExtension As String, ByVal Filter
    As String, ByRef FilterIndex As Long, ByVal InitialDir As String, ByVal DialogTitle As String,
    ByRef Flags As EFileDialogFlags) As String
136 ' Implementation of ShowFileOpenDialog (see it for description)
137
138 Dim OFN As OpenFilename ' Structure with dialog information
139 Dim Result As Long ' API return value
140 Dim indNull As Long ' Index of Null character (ASCII 0)
141
142 Dim ErrMsg As String, ErrCode As Long ' Error information
143
144 ' Populate OFN structure with dialog initialization information

```

```

145 With OFN
146 .lStructSize = Len(OFN) ' Length of structure
147 .hwndOwner = hwndOwner ' Owner window handle
148 .lpstrFilter = Replace(Filter, "|", vbNullChar) & vbNullChar & vbNullChar ' Filters
149 .nFilterIndex = FilterIndex ' Selected filter
150 .lpstrFile = String(MAXFILE, 0) ' Filename
151 .nMaxFile = MAXFILE ' Max length of filename
152
153 ' Initial directory
154 If LenB(InitialDir) = 0 Then
155 .lpstrInitialDir = CurDir ' No initial directory given, use current directory
156 Else
157 .lpstrInitialDir = InitialDir ' Use given initial directory
158 End If
159 .lpstrTitle = DialogTitle ' Title of dialog
160 .Flags = Flags ' Various flags passed by caller
161 .lpstrDefExt = DefaultExtension ' Default file extension
162 End With
163
164 ' Show the Open dialog
165 Result = GetOpenFileNameA(OFN)
166
167 If Result <> 0 Then
168 ' Success
169 With OFN
170 ' Retrieve selected filename and set it as the function return value
171 indNull = InStr(.lpstrFile, vbNullChar)
172 If indNull > 0 Then
173 FileDialog = Left$(.lpstrFile, indNull - 1) ' Get rid of terminating null character
174 Else
175 FileDialog = .lpstrFile ' No terminating null
176 End If
177
178 Flags = .Flags ' Return flags to caller - they may have changed
179 FilterIndex = .nFilterIndex ' Return selected filter to caller
180 End With
181 Else
182 ' Error or cancel
183 ' Call ComDlgExtendedError to get error code
184 ErrCode = ComDlgExtendedError()
185
186 ' Select error message
187 Select Case ErrCode
188 Case 0
189 ' Cancel pressed, no error
190 Case CDERR_DIALOGFAILURE
191 ErrMsg = "Dialog box could not be created."
192 Case CDERR_FINDRESFAILURE
193 ErrMsg = "Failed to find a resource."
194 Case CDERR_NOINSTANCE
195 ErrMsg = "Instance handle missing."
196 Case CDERR_INITIALIZATION
197 ErrMsg = "Failure during initialization. Possibly out of memory."
198 Case CDERR_NOHOOK

```

```
199     ErrMsg = "Hook procedure missing."
200 Case CDERR_LOCKRESFAILURE
201     ErrMsg = "Failed to lock a resource."
202 Case CDERR_NOTEMPLATE
203     ErrMsg = "Template missing."
204 Case CDERR_LOADRESFAILURE
205     ErrMsg = "Failed to load a resource."
206 Case CDERR_STRUCTSIZE
207     ErrMsg = "Internal error - invalid struct size."
208 Case CDERR_LOADSTRFAILURE
209     ErrMsg = "Failed to load a string."
210 Case FNERR_BUFFERTOOSMALL
211     ErrMsg = "File name buffer too small"
212 Case CDERR_MEMALLOCFailure
213     ErrMsg = "Unable to allocate memory for internal dialog structures."
214 Case FNERR_INVALIDFILENAME
215     ErrMsg = "Invalid file name."
216 Case CDERR_MEMLOCKFAILURE
217     ErrMsg = "Unable to lock memory."
218 Case FNERR_SUBCLASSFAILURE
219     ErrMsg = "Subclass failure, out of memory."
220 Case Else
221     ErrMsg = "Unknown error."
222 End Select
223 If LenB(ErrMsg) Then
224     ' An error occurred. Display it.
225     If LenB(DialogTitle) Then
226         ErrMsg = "Error #" & ErrCode & " with dialog " & DialogTitle & vbCrLf & ErrMsg
227     Else
228         ErrMsg = "Error #" & ErrCode & " with file open dialog" & vbCrLf & ErrMsg
229     End If
230     MsgBox ErrMsg, vbExclamation
231 End If
232 End If
233
234 End Function
235
236 Sub Main()
237     ' PicInfo main procedure
238     ' The program starts here
239
240     ' Store program title in global variable
241     ProgramTitle = App.Title
242
243     ' Show main form
244     PicForm.Show
245
246 End Sub
247
248
```