SaveAs Dialog **Sample Help**

Sample Description: SaveAs Dialog

Points of Interest

- Adding a SaveAs Dialog to your Application
- Posting the SaveAs Dialog
- Selecting a File
- Applying a Selected File

Control

SaveAsDialog

For Help on Help, Press F1

SaveAs Dialog

The SaveAs Dialog permits selection of a drive, directory, and name for a file that the user wants to specify for saving application information. The default caption for the title bar is "Save As," however this can be changed by setting the SaveAsDialog's **Title** property before the SaveAsDialog is posted.

Before displaying the SaveAsDialog, you can set various options through the SaveAsDialog's properties that restrict the functionality of the SaveAsDialog. The following describes some of the available property options:

Property	Behavior
Filter	Specifies what types of files are displayed in the File list box. By default, no filter is set. If you don't specify the Filter property, the user must explicitly type a file name in the File Name edit box. The user cannot click a name from a list of files filtered to the File list box.
NoNetworkButton	Displays the Network button if True, otherwise hides it.
NoChangeDir	Does not permit changing directories if this property is True.

To run the SaveAsDialog sample:

1. Enter a desired text string in the Title box
2. Then click on the various options and checkboxes displayed in the sample form. This sets the various properties available in the SaveAsDialog.
3. Now click the **SaveAs** button and select a file from the SaveAsDialog.
4. Click the **OK** button on the SaveAsDialog. Information obtained from the SaveAsDialog properties is displayed at the bottom of the sample form.

Adding a SaveAs Dialog to your Application

A SaveAs dialog can easily be added to your application for selecting files. This is accomplished by embedding a SaveAsDialog as an object in your form's property list. In the example shown, the SaveAsDialog was embedded and given a name "SaveAsPanel."

When added to an application, the user can select a filename from various drives and directories. If the Network button option is used, the user may select files from other systems on the network.

Once a file has been selected, information about the file is loaded into the SaveAsDialog's various properties. This information may be useful for determining the type of action you wish to take in your program code.

Posting the SaveAs Dialog

To post a SaveAs Dialog, you may first wish to place an informative label in the title area of the SaveAs Dialog, such as "Please select filename." This can be done by setting the SaveAs Dialog Title property. For example: SaveAsPanel.Title = "Please select filename".

In addition, you may want to limit the files being displayed in the file list box by setting the filter property. For example: SaveAsPanel.Filter = "Bitmap files (*.bmp)|*.bmp|" Filters may incorporate multiple file type extensions, thereby allowing more than one type of file to be presented. For example: SaveAsPanel.Filter = "System files (*.sys)|*.sys|Config files (*.ini)|*.ini|"

To actually post the SaveAsDialog, you need only run the dialog's **Execute** method. For example: SaveAsPanel.Execute

If you examine the code found in the **btnSaveAs_Click** method, you will see that once the Execute method is issued, your program code is temporarily suspended until either the **OK** or **Cancel** button on the SaveAs dialog is clicked.

Selecting a File

Once a SaveAs Dialog is posted, you may select a file by clicking on one of the displayed filenames. The user may change drives by selecting a drive from the drive combobox. The user can also change directories by clicking on various directory names and clicking the **OK** button. It is important to note that clicking OK on a filename will accept the filename, however clicking OK on a directory simply changes the currently displayed directory.

Once a filename has been selected, information such as file size, date created, etc. are recorded in the SaveAsDialog's properties. This information may be useful to your program. Should the user decide to abort out of the file selection operation, they can click the **Cancel** button.

Applying a Selected File

Once a filename has been selected, you may want your program code to carry out a specific operation on that filename, such as opening the file for editing, printing the file, merging the file into your application, etc. In the sample demonstration, only the information about the file that was selected in the SaveAs Dialog is displayed.

The technique for determining the selected directory and filename is different than that used for the Open Dialog sample. The following EchoDirectory and EchoFilename functions are used to separate the filename and directory names from the selected file. This is not the recommended approach for separating filenames and directory names from filepaths. But the example does show how the **Instr** command can be used with some application logic to parse specific information out of a string.

The following EchoDirectory function takes the entire filepath from the calling routine and returns only the directory portion by parsing off the filename and the last "\" character.

```
Function EchoDirectory(filepath As String) As String
   Dim found As Integer
   Dim startpos As Integer

   If filepath == "" Then
      EchoDirectory = ""
   Else
      ' Locate the last "\" in the filepath string
      startpos = 0
      found = 1
      Do While found <> 0
         found = Instr(startpos, filepath, "\")
         If found <> 0 Then
            startpos = found + 1
         Else
            If startpos == 0 Then
               EchoDirectory = filepath
            Else
               startpos = startpos - 2
               EchoDirectory = Left(filepath, startpos)
            End If
         End If
      Loop
   End If
End Function
```

The following EchoFilename function takes the entire filepath from the calling routine and returns only the filename portion by parsing off the filename from the end of the filepath.

```
Function EchoFilename(filepath As String) As String
   Dim found As Integer
   Dim startpos As Integer
   Dim length As Integer

   If filepath == "" Then
      EchoFilename = ""
   Else
      ' Locate the last "\" in the filepath string
      startpos = 0
      found = 1
      length = Len(filepath)
      Do While found <> 0
         found = Instr(startpos, filepath, "\")
         If found <> 0 Then
            startpos = found + 1
         Else
            If startpos == 0 Then
               EchoFilename = filepath
            Else
               startpos = startpos
```

```
          EchoFilename = Mid(filepath, startpos, (length - startpos) + 1)
        End If
      End If
    Loop
  End If
End Function
```

Neither of these routines is a recommended approach to determining directory and filenames. A much better approach was shown with the OpenDialog as shown below:

```
If OpenPanel.FileName <> "" Then
    lblFilePath.Caption = OpenPanel.FileName
    FileObject.FileName = OpenPanel.FileName
    ' Strip off trailing \
    length = Len(FileObject.Path)
    temp = Left$(FileObject.Path, length - 1)
    lblDirectory.Caption = temp
    lblFileName.Caption = FileObject.Name
    lblFileSize.Caption = FileObject.Size
    lblFileExtension.Caption = FileObject.Extension
    lblFileDate.Caption = FileObject.Date
    lblFileTime.Caption = FileObject.Time
End If
```

This technique uses a FileObject, which when handed a filepath, automatically sets corresponding properities for both filename and directory.

Once the SaveAs Dialog has been used to select a filename, your program code should take the corresponding properties and perform some type of save function.