Main Menu

Please read this help file thoroughly! There is lots of important and updated information in the Tech Note, Getting Started, What's New, and How To's sections.

Overview

ActiveZipper Pro is a feature rich File Compression control that contains many features that other compression components do not. The control is supported in a variety of ActiveX Control Containers such as Visual Basic 5.0, 6.0, Microsoft Access, Excel, FoxPro, and many others. It is also lightweight, streamlined for memory requirements, and optimized for speed. You can distribute the registered version without adding extra overhead to your programs while adding vital compression features. If by any chance you need a custom version of ActiveZipper Pro, the source code is available for purchase.

ActiveZipper Pro supports both the native compression type (.az) and the widely available PKzip/PKunzip format (.zip). Both interfaces are easy to use and a breeze to implement in your applications. Each interface has a multitude of features that makes manipulating compressed archives very simple and quick. To make things easier, features already available in ActiveZipper Pro such as Disk Spanning can be used with the PKzip/PKunzip implementation.

Limitation in Shareware Version

- When using PKzip/PKunzip you can only compress or decompress 6 files to/from the archive.

Registered versions are not limited in any way.

What's New

ActiveZipper Pro 1.0f 1096

- More bugs with splitting files and assembling them have been addressed with the inclusion of two new Error Codes. There also seems to be some confusion regarding the ControlStatus event and that has been fixed as well. I am also looking for some beta testers to assure higher quality releases in the future. Please email me at sk@iessoft.com with your name, past testing experience, and the version(s) of Visual Basic you work with regularly.

ActiveZipper Pro 1.0f 1094r3

- Some more bugs eliminated in the PKzip/PKunzip portion of ActiveZipper Pro. Some typographical errors also corrected in the help file.

ActiveZipper Pro 1.0f 1094r2

- A nasty bug caused people not able to unzip files that were PKzipped. Eliminated.

ActiveZipper Pro 1.0f 1094

- ActiveZipper now supports PKzip/PKunzip files.
- Two new dependancies for ActiveZipper Pro: Zip32.dll and unZip32.dll. Please see the Tech Note for more information.
- This completely redesigned and rewritten help file.
- The internal use of functions, subs, and methods has been streamlined to make readability and maintenance easier and also to facilitate the ability to release future revisions faster.
- The default value for Level has been changed to 256.
- InternetServer Property has been removed due to the Progress Event being added.
- The UsingClass property has been removed.
- The Progress Property has been removed.
- Progress Property has been changed to an Event. Monitoring progress is less cumbersome and it does away with the InternetServer property. The ControlStatus property also lets you monitor the progress of zipping and unzipping ZIP files to some degree. Please see the Tech Note for more information.

What's Coming

- Complete support for PKunzip/PKzip files. Also, more properties and methods will be added for the manipulation of zip archives. Registered users will automatically receive the upgrades for them along with updated documentation.

Technical Support

Registered users can receive unlimited technical support by emailing sk@iessoft.com. Users with the shareware version you can email bug reports, technical support, however will be answered but please be patient as registered users come first.

You can also post questions on the website at: http://www.iessoft.com

Tech Note

Distribution

Credits

Special thanks goes to *Craig Gill* who has spent a lot of time with me creating ActiveZipper Pro, testing it, and giving insight on any problems that arose during the creation of this component. You can visit his site here: http://web2.airmail.net/cgill/vb.htm

Another person who has helped a great deal is *Siegfried Rings*. He helped me come up with the original idea and gave me lots of information on how to start creating ActiveX controls in Visual Basic.

Finally, last but not least, *Kevin Lingofelter* has helped me with the implementation of PKzip/PKunzip and offered a lot of help on getting around a few of Visual Basic's limitations.

A big thank you goes to each and every one of them for helping me out!

Release History

October 12, 1997 - ActiveZipper Pro officially released. *v1.0f Build 1080*

October 20, 1997 - A few bugs caused people not to be able to decompress archives. *v1.0f Build 1082*

November 12, 1997 - This version is much more stable, many little bugs stomped. *v1.0f Build 1084*

November 16, 1997 - Major upgrade! CRC Checksum added to ensure integrity of individual files and many bugs fixed. Many new methods added to make control easier to use, but it is still backwards compatible. Next update will (or should) include disk spanning! *v1.0f Build 1086*

November 18, 1997 - Adding features introduced a few bugs but they were fixed in this version! Look for a new VB Example to go along with ActiveZipper Pro in a few days! *v1.0f Build 1088*

November 29, 1997 - After listening to a few users, this control was redone a little under the hood to make it more programmer freindly and now comes with a brand new demo/example! *v1.0f Build 1090*

February 1, 1998 - Added the option of encrypting files. Only one property added to do so. No properties need to be set to decompress an encrypted archive, all done automatically. Disk spanning was also added to this release! *v1.0f Build 1092*

March 12, 1998 - ActiveZipper Pro has a real installer, this was due to the fact that the control did not want to register itself properly on some machines, hopefully this will alleviate the problems people are having. *v1.0f Build 1092r2*

April 14, 1998 - This release includes beta support for Internet Information Server (3.0 and 4.0). This is also the last time I will switch installers and I'm currently using VB5's Setup Wizard. *v1.0f Build 1092r3*

October 28, 1998 - General maintainence release. Some minor bugs squashed and tweaked the compression engine slightly. *v1.0f Build 1092r4*

November 25, 1998 - Added support for PKzip/PKunzip. This release included some minor interface changes in the control and a redesigned help file. *v1.0f Build 1094*

Janurary 4, 1999 - First release of the new year and this version eliminates a nasty bug in the UnZipIt method (thanks to a hawk eyed user). *v1.0f Build 1094r2*

January 12, 1999 - Some more bugs found in the PKzip/PKunzip portion of ActiveZipper Pro, these have been fixed along with some typographical errors corrected in the help file. *v1.0f 1094r3*

February 13, 1999 - Fixed problems with the Splitting and Assembling of files and cleared up the confusion of the ControlStatus/Progress event. Three new error codes introduced as well.

Getting Started

Minumum Requirements:
Visual Basic 5.0 with Service Pack 2 installed
Regsvr32.exe. Visual Basic 6.0 also supported. But not tested, I have VB6 but still see it as too buggy to port it over. Perhaps by SP3 for VB6 I will then convert it.

Any environment that can host ActiveX controls. This means you must also distribute the runtime file(s).

Recommended:
Visual Basic 5.0 with Service Pack 3 installed. (Get it here)

Please note: If you are unsure of what patched version of Visual Basic 5.0 you are running then check the About Box. If it says SP2 anywhere you can still use the control but from what I see Service Pack 3 is more stable.

Installation:
Please note: If you have a previous copy of ActiveZipper Pro and you cannot uninstall it via the Add/Remove applet in the Control Panel you must unregister it manually. Like this: *regsvr32 ActiveZipperproS.ocx /u* (or 8.3 equivilent) You can find Regsvr32.exe on your VB5 CD.

If you have the shareware version and purchased the registered version please uninstall the shareware version using the Add/Remove applet in the Control Panel and you can proceed to install the registered version. The information given above is in case you cannot uninstall the OCX usint the Add/Remove applet in the Control Panel.

Quick Code NOT USING PKZIP/PKUNZIP:
```
'To compress a few files:
'This requires 1 label for progress and 'preferrebly put this code in a
command button 'or anything similar
Private Sub ActiveZipperPro1_ControlStatus(Val As String, ByVal ZipInf As
String)
Label1.Refresh
Label1.Caption = Val
End Sub

Private Sub Command1_Click()
ActiveZipperPro1.SourceFile = "c:\myfile.exe,c:\test.exe"
ActiveZipperPro1.OutPutFile = "archivetest"

ActiveZipperPro1.Destination = "c:\mydirectory\" 'dont worry, it will 'create
it for you!
ActiveZipperPro1.Compress
'-- End Compress
End Sub

'To Decompress from the same archive as above
'This requires 1 label for progress and 'preferrebly put this code in a
command button or 'anything similar

Private Sub ActiveZipperPro1_ControlStatus(Val As String, ByVal ZipInf As
String)
Label1.Refresh
Label1.Caption = Val
End Sub

Private Sub Command2_Click()
```

```
Dim dummy As String
ActiveZipperPro1.SourceFile = "c:\archivetest.az"
dummy = ActiveZipperPro1.Ratio(1) 'To init the archive, very 'important!
ActiveZipperPro1.Destination = "c:\yourdir\"
ActiveZipperPro1.DecompressedFile = "myfile.exe"
ActiveZipperPro1.Decompress 1, True, 1 'To decompress the first file in the
'archive and run the file in a normal window
'-- End Decompress
End Sub
```

Quick Code USING PKZIP/PKUNZIP:

```
'-- Begin Zip
'To ZIP a few files:
'This requires 1 label for progress and 'preferrebly put this code in a
command button 'or anything similar
Private Sub ActiveZipperPro1_ControlStatus(Val As String, ByVal ZipInf As
String)
Label1.Caption = ZipInf
End Sub

Private Sub Command1_Click()
ActiveZipperPro1.SourceFile = "c:\ham.exe,c:\fan.exe"

ActiveZipperPro1.OutPutFile = "C:\myfile.zip" 'Very important.. when creating
ZIP files you MUST include a path and/or directory name.
ActiveZipperPro1.ZipIt True, False, False, vbNullstring
'-- End Zip
End Sub

'-- Begin Unzip
'To Decompress from the same archive as above
'This requires 1 label for progress and 'preferrebly put this code in a
command button or 'anything similar

Private Sub ActiveZipperPro1_ControlStatus(Val As String, ByVal ZipInf As
String)
Label1.Caption = ZipInf
End Sub

Private Sub Command2_Click()
Dim dummy As String
ActiveZipperPro1.SourceFile = "c:\archive.zip"
'You can tell ActiveZipperPro to unzip all of the files by leaving
the .OutPutFiles property empty or you can tell it to unzip certain files by
using a combination of methods and properties. See zMaxFiles and zFileName. I
will show you the latter in the How To's because it is harder to implement.
ActiveZipperPro1.UnZipIt True 'Keep directory structure
'-- End Unzip
End Sub
```

List of Properties

CopyFiles

DecompressedFile

Destination

DiskDrive

DiskNumber

DiskSize

Extension

Garble

Level

MaxFiles

OutputFile

Password

Protected

SourceFile

StopOperation

Version

Web

zFileDate

zFileSize

CopyFiles Property

Syntax:

*object.***CopyFiles**

Remarks:

Set to **True** if you would like ActiveZipper Pro to copy the split files to disk. Please note that it is necessary to put a *Message Box* of some sort in the NextDisk event to notify the user to insert the next disk. The only accepted values are **True** or **False**.

Example:

```
ActiveZipperPro1.CopyFiles = True
```

DecompressedFile Property

Syntax:

*object.***DecompressedFile**

Remarks:

Name of file to be decompressed to. You only have to enter the filename, no directory or drive is needed.

Example:

```
ActiveZipperPro1.DecompressedFile = "thisfile.exe"
```

Destination Property

Syntax:

object.**Destination**

Remarks:

Tells ActiveZipper where to compress or decompress. This property is automatically used in conjunction with OutPutFile and DecompressedFile. When using the PKzip/PKunzip portion of the control the .**Destination** property tells ActiveZipper Pro where to Zip/Unzip archives. If the directory you specify does not exist, ActiveZipper Pro will create it for you.

Example:

```
ActiveZipperPro1.Destination = "c:\dos"
```

DiskDrive Property

Syntax:

object.**DiskDrive**

Remarks:

The default is the A:\ drive. You can enter any other drive or drive\directory combination

Example:

```
ActiveZipperPro1.DiskDrive = "B:\"
```

or

```
ActiveZipperPro1.DiskDrive = "B:\target\"
```

DiskSize Property

Syntax:

object.**DiskSize**

Remarks:

If you wish to copy the split files to a disk you should specify what size to split the archive to. The accepted values are:

1.4 (Default)
1.2
720
360

Example:

```
ActiveZipperPro1.DiskSize = "1.4"
```

Extension Property

Syntax:

object.**Extension**

Remarks:

Enter a 3 letter or number string for this property. Please note that it is necessary to enter a period before the extension. Leave blank if you wish to use the default extension. The default is "**.az**" This property is not valid for PKzip/PKunzip operations.

Example:

```
ActiveZipperPro1.Extension = ".azp"
```

Garble Property

Syntax:

object.**Garble**

Remarks:

Set to **True** if you wish to encrypt the files before adding them to an archive. By default this property is set to **False**. This property is not valid for PKzip/PKunzip operations.

Example:

```
ActiveZipperPro1.Garble = True
```

Level Property

Syntax:

object.**Level**

Remarks:

Use values between **256** and **3065**. The higher the number, the better the compression and the longer it takes to compress.However, it will yield a better compression ratio. It is not necessary to set the Level when decompressing an archive (or any files), ActiveZipper Pro will automatically detects the compression level used. The default is **256**. This property is not valid for PKzip/PKunzip operations.

Example:

```
ActiveZipperPro1.Level = 512
```

MaxFiles Property

Syntax:

object.**MaxFiles**

Remarks:

Tells you how many files are in the current archive. The only prerequisite to using this property is that you must call *object*.**Ratio**(1). Otherwise you will get an error. This is a Read-Only property and is not valid for PKzip/PKunzip operations.

Example:

```
'To initialize the archive
ActiveZipperPro1.SourceFile = "c:\myarchive.az"
ActiveZipperPro1.Ratio(1)
Label1.Caption = "Number of files in archive: " & ActiveZipperPro1.MaxFiles
```

OutputFile Property

Syntax:

object.**OutPutFile**

Remarks:

Name of the file to be compressed to. When decompressing archives, use the DecompressedFile property. Notice that no extension is needed. The use of this property changes when using the PKunzip portion. If .**OutputFile** is empty then ActiveZipper Pro will unzip all of the files in the archive. To fill the .**OutPutFile** property you should check these two methods out: zMaxFiles and zFileName. For zipping files with PKzip you (for now) are required to input the whole path and filename of the archive you want to create.

Example not using PKunzip:

```
ActiveZipperPro1.OutputFile = "myarchive"
```

Example using PKzip:

```
ActiveZipperPro1.OutPutFile = "c:\myzipfile.zip"
```

Example using PKunzip:

```
ActiveZipperPro1.OutputFile = "setup.exe,car.bmp"
```

Password Property

See Also: Protected

Syntax:

object.**Password**

Remarks:

Enter a string to protect an archive. Leave blank for none. To make sure that no errors are raised when decompressing archives it is a good idea to check to see if the .**Protected** property of an archive is not **True**. If the password specified in this property is incorrect an error will be raised. This property is not valid for PKzip/PKunzip operations.

Example:

```
ActiveZipperPro1.Password = "mypassword"
```

Protected Property

Syntax:

object.**Protected**

Remarks:

A Read-Only property that will tell you if the current archive is password protected. If it is and you try to decompress without supplying a password (or a correct one) it will raise an error. This property is not valid for PKzip/PKunzip operations.

Example:

```
Label1.Caption = "Archive protected: " & ActiveZipperPro1.Protected
```

SourceFile Property

Syntax:

object.**SourceFile**

Remarks:

This property has three functions:
One for specifying files to be compressed into an archive.
The second one for entering the archive location for decompressing, finally the third function is for using the PKzip portion of the control to ZIP files.

Example for compressing with or without PKzip:

```
ActiveZipperPro1.SourceFile = "c:\myfile.bmp,c:\file.exe"
```

Example for decompressing not using PKunzip:

```
ActiveZipperPro1.SourceFile = "C:\myarchive.az"
```

StopOperation Property

Syntax:

object.**StopOperation**

Remarks:

Set this property to **True** if you wish to interrupt the current operation. Remember to reset this property to false before compressing or decompressing. The default value is **False**. This property is not valid for PKzip/PKunzip operations.

Example:

```
ActiveZipperPro1.StopOperation = True
```

Version Property

Syntax:

object.**Version**

Remarks:

Gives current version of the control.

Example:

```
Label1.Caption = "Current Version is: " & ActiveZipperPro1.Version
```

Web Property

Syntax:

object.**Web**

Remarks:

When set to **True** ActiveZipper Pro will not overwrite critical system files. When setting it to **False** please be wary of not overwriting important files on the user's system. This property is not valid for PKzip/PKunzip operations. This property is not valid for PKzip/PKunzip operations. However, it may be implemented in the future.

Example:

```
ActiveZipperPro1.Web = True
```

List of Methods

List of How To's

How to: Compress all of the files in one directory using the File List Box

How to: Unzip selected files in a List Box

About Method

Syntax:

object.**About**()

Description:

Shows the About Box for ActiveZipper Pro

Example:

```
ActiveZipperPro1.About
```

AssembleFiles Property

Syntax:

object.**AssembleFiles**(*ArcFile*, *SplitLoc*)

Description:

Creates the original file taken from split files.

ArcFile - A string that contains the arc.id file, a simple header file that gives a little information about the split files.
SplitLoc - A string that contains the directory where the split files (*.0??) reside.

After calling this method the original file will be created and you can manipulate it via ActiveZipper Pro.

Example:

```
ActiveZipperPro1.AssembleFiles "c:\arc.id", "c:\test"
```

Compress Method

Syntax:

object.**Compress**()

Description:

Starts compressing files, ActiveZipper Pro automatically makes sure that it does not start another operation until it has finished its first task. This does not apply to PKzip.

Example:

```
ActiveZipperPro1.SourceFile = "c:\test.exe,c:\swap.dll,c:\info.exe"
ActiveZipperPro1.OutPutFile = "myarchive"
ActiveZipperPro1.Destination = "c:\mydirectory"
ActiveZipperPro1.Compress
```

Distribution

You are more than welcome to distribute the SHAREWARE version of the control only if the original archive (azpzip.zip) is intact and nothing was modified. You may collect a nominal fee for distribution purposes (such as for disks, CDROMs, etc.)

Registered Users: Please read the terms in the License Agreement that was installed along with ActiveZipper Pro to get the details on distribution.

Tech Note

Please note that the .**Progress**, .**InternetServer**, and .**UsingClass** properties has been removed. They have been taken out because of the addition of a new Event called ControlStatus. ControlStatus can easily monitor the progress of compressing or decompressing archives without having to put in *Set Label1* statements everywhere and still have it work on IIS 3.0 and 4.0. To properly show progress when **not** using the PKzip/PKunzip portion of ActiveZipper Pro, for example using a Label control the Sub would look like this:

```
Private Sub ActiveZipperPro1_ControlStatus(Val As String, ByVal ZipInf As
String)
Label1.Refresh
Label1.Caption = Val
End Sub
```

To show progress when using the PKzip/PKunzip portion of ActiveZipper Pro you would use the same procedure as above except it will look like this:

```
Private Sub ActiveZipperPro1_ControlStatus(Val As String, ByVal ZipInf As
String)
Label1.Caption = ZipInf
End Sub
```

Now that PKzip/PKunzip has been added to ActiveZipper Pro there are two more dependencies: **zip32.dll** and **unzip32.dll**. When distributing make sure that the two DLLs go to the Windows/System (or WINNT/SYSTEM32 if your program is going to run on Windows NT) directory of that computer. If the DLLs are not present in either of those directories your program will fail to function or load properly.

CRCcheck Method

Syntax:

object.**CRCcheck**(*FileRecNumber*)

Description:

FileRecNumber – You must first specify a valid archive in the sourcefile. Gives CRC check for single file in the archive only not the whole archive. This method is not valid for PKzip/PKunzip operations.

Example:

```
ActiveZipperPro1.SourceFile = "c:\myarchive.az"
Label1.Caption = ActiveZipperPro1.CRCcheck(1) '1st file in archive.
Call .MaxFiles right after .CRCcheck(1) if you are unsure of how many more
files there are in the archive.
```

Decompress Method

Syntax:

object.**Decompress**(*RecNum*, *RunFile*, *SreenState*)

Description:

RecNum - The index of the compressed file in the archive. To decompress the first file you would use the number 1 and so on.
RunFile - A Boolean property where you specify True if you wish the file to be run with it's appropriate program.
ScreenState - Use either 0 for minimized, 1 for normal, 2 for maximized.
This method is not valid for PKzip/PKunzip operations.

Example:

```
ActiveZipperPro1.SourceFile = "c:\myarchive.az"
ActiveZipperPro1.Decompress(1, True, 1)
```

FileDate Method

Syntax:

object.**FileDate**(*FileRecNumber*)

Description:

FileRecNumber – An integer, you enter the a valid index to find the file's date of an archive. This method is not valid for PKzip/PKunzip operations.

Example:

```
ActiveZipperPro1.SourceFile = "c:\myarchive.az"
Label1.Caption = ActiveZipperPro1.FileDate(1) '1 is the first file in the
archive. Call .MaxFiles right after .FileDate(1) if you are unsure of how
many more files there are in the archive.
```

PackedSize Method

Syntax:

object.**PackedSize**(*FileRecNumber*)

Description:

FileRecNumber- Returns the compressed size of the file in the archive.
This method is not valid for PKzip/PKunzip operations.

Example:

```
ActiveZipperPro1.SourceFile = "c:\myarchive.az"
Label1.Caption = ActiveZipperPro1.PackedSize(1) '1 is the first file in the
archive. Call .MaxFiles right after .PackedSize(1) if you are unsure of how
many more files there are in the archive.
```

Ratio Method

Syntax:

object.**Ratio**(*FileRecNumber*)

Description:
FileRecNumber – Returns a value telling you how much the file is compressed versus the original. This method is not valid for PKzip/PKunzip operations.

Example:

```
ActiveZipperPro1.SourceFile = "c:\myarchive.az"
Label1.Caption = ActiveZipperPro1.Ratio(1) '1 is the first file in the
archive. Call .MaxFiles right after .Ratio(1) if you are unsure of how many
more files there are in the archive.
```

UnPackedSize Method

Syntax:

object.**UnPackedSize**(*FileRecNumber*)

FileRecNumber – Returns the uncompressed (original) file size in the archive. This method is not valid for PKzip/PKunzip operations.

Example:

```
ActiveZipperPro1.SourceFile = "c:\myarchive.az"
Label1.Caption = ActiveZipperPro1.UnPackedSize(1) '1 is the first file in the
archive. Call .MaxFiles right after .UnPackedSize(1) if you are unsure of how
many more files there are in the archive.
```

Garbled Method

Syntax:

object.**Garbled**(*FileRecNumber*)

Description:

FileRecNumber - Returns a boolean value that tells you if the files inside the archive are encrypted or not. You do not need to run this check on every file because all of them will be encrypted. This method is not valid for PKzip/PKunzip operations.

Example:

```
ActiveZipperPro1.SourceFile = "c:\myarchive.az"
Label1.Caption = ActiveZipperPro1.Garbled(1) '1 is the first file in the
archive. Call .MaxFiles right after .Garbled(1) if you are unsure of how many
more files there are in the archive.
```

List of Events

Completed

ControlStatus

GetDiskNumber

NextDisk

Completed Event

Syntax:

Private Sub *object*_**Completed**()

Description:

Fires every time ActiveZipper Pro has finished compressing or decompressing files. That goes the same for zipping and unzipping with PKzip/PKunzip.

Example:

```
Private Sub ActiveZipperPro1_Completed()
Msgbox "Finished!"
End Sub
```

ControlStatus Event

Syntax:

Private Sub *object*_**ControlStatus**(*Val* **As String**, **ByVal** *ZipInf* **As String**)

Description:

This event monitors native file compression and decompression. It also gives a good amount of feedback on the progress of zipping and unzipping with PKzip/PKunzip.

Example for compressing/decompressing WITHOUT PKzip/PKunzip:

```
Private Sub ActiveZipperPro1_ControlStatus(Val As String, ByVal ZipInf As
String)
Label1.Refresh
Label1.Caption = Val
End Sub
```

Example for zipping/unzipping WITH PKzip/PKunzip:

```
Private Sub ActiveZipperPro1_ControlStatus(Val As String, ByVal ZipInf As
String)
Label1.Refresh
Label1.Caption = ZipInf
End Sub
```

GetDiskNumber Event

Syntax:

Private Sub *object*_**GetDiskNumber**()

Description:

Fires when the DiskNumber Property has changed. Useful to alert the user to get ready to copy disks if you wish to let them know.

Example:

```
Private Sub ActiveZipperPro1_GetDiskNumber()
Msgbox "Please get ready to insert disk number #" &
ActiveZipperPro1.DiskNumber
End Sub
```

DiskNumber Property

Syntax:

object.**DiskNumber**

Remarks:

A Read-Only property that will tell you regardless of whether or not the .CopyFiles property is set to **True** or not the number of disks a split archive would require.

Example:

```
Label1.Caption = "Number of disks currently required: " &
ActiveZipperPro1.DiskNumber
```

OR

```
Private Sub ActiveZipperPro1_GetDiskNumber()
Msgbox "Please get ready to insert disk number #" &
ActiveZipperPro1.DiskNumber
End Sub
```

NextDisk Event

Syntax:

Private Sub *object*_**NextDisk**()

Description:

Fires just before ActiveZipper Pro attempts to copy another file to a disk. The CopyFiles property must be set to **True** for the event to work.

Example:

```
Private Sub ActiveZipperPro1_NextDisk()
Msgbox "Please insert disk number #" & ActiveZipperPro1.DiskNumber
End Sub
```

UnZipIt Method

See Also: zMaxFiles, zFileName

Syntax:

object.**UnzipIt**(*KeepDirs*)

Description:

Unzips all of or some of the files. All files unzipped will overwrite any files of the same name that exist.

KeepDirs - A boolean value that when set to **True** will unzip keeping the directory structure of the files inside.

Example:

```
ActiveZipperPro1.SourceFile = "c:\somefile.zip"
ActiveZipperPro1.Destination = "c:\mydir"
ActiveZipperPro1.OutPutFile = "test.jpg,test.exe" 'If you do not specify any
files to be unzipping in the OutPutFile property, ActiveZipper Pro will unzip
all of the files in the directory.
ActiveZipperPro1.UnZipIt True
```

ZipIt Method

See Also: SourceFile, OutPutFile

Syntax:

object.**ZipIt**(*UsePath, RecurseDir, Update, BasePath*)

Description:

UsePath - A boolean value that when set to **True** will keep the pathnames of the files that are going to be zipped.

RecurseDir - A boolean value that when set to True will check the .SourceFile property of a value such as **.Exe, C:\Test\*.Bat*.

Update - When set to **True** it will overwrite the files in the archive if the file is newer than the one present (if the archive exists). Othewise if set to **False** then it will overwrite them.

BasePath - Overrides the pathname from the files in the .SourceFile property with the one you specify. Either set it to "" or vbNullString if you do not wish to override it.

Example:

```
ActiveZipperPro1.SourceFile = "c:\temp\test.exe,c:\red.txt"
ActiveZipperPro1.OutPutFile = "c:\test\myzipfile.zip"
ActiveZipperPro1.ZipIt True, False, True, vbNullString
```

zFileDate Property

See Also: zMaxFiles, zFileName

Syntax:

object.**zFileDate**

Remarks:

Reports the file date/time of a file inside a zipped archive. The prerequisite to using this property is that zFileName must be called before accessing this property.

Example:

```
ActiveZipperPro1.SourceFile = "c:\myarchive.zip"
ActiveZipperPro1.zMaxFiles
ActiveZipperPro1.zFileName(1) '2nd file in archive, 0 is the first file in
the archive.
Label1.Caption = "Date : " & ActiveZipperPro1.zFileDate
```

zFileSize Property

See Also: zMaxFiles, zFileName

ect.**zFileSize**

Remarks:

Reports the original file size of a file inside a zipped archive. The prerequisite to using this property is that zFileName must be called before accessing this property.

Example:

```
ActiveZipperPro1.SourceFile = "c:\myarchive.zip"
ActiveZipperPro1.zMaxFiles
ActiveZipperPro1.zFileName(1) '2nd file in archive, 0 is the first file in
the archive.
Label1.Caption = "Size : " & ActiveZipperPro1.zFileSize
```

zFileName Method

See Also: zFileDate, zFileSize, zMaxFiles
Syntax:

object.**zFileName**(*Rec*)

Description:

Returns a string of the filename from the zipped archive based on the *Rec* value. After this method has been called the properties zFileSize and zFileDate will be filled with the appropriate information regarding that file.

Rec - A string which can be inputted with a valid number from 0 to the value returned from zMaxFiles to get a filename based on the argument given.

Example:

```
ActiveZipperPro1.SourceFile = "c:\myarchive.zip"
Label1.Caption = "Number of files: " & ActiveZipperPro1.zMaxFiles
Label2.Caption = ActiveZipperPro1.zFileName(1) '2nd file in archive, 0 is the
first file in the archive.
```

zMaxFiles Method

See Also: UnZipIt, zFileDate, zFileName, zFileSize

Syntax:

object.**zMaxFiles**()

Description:

Returns the number of files within a zipped archive. This method is used in conjunction with zFileName, zFileDate, and zFileSize to find out more information on each file in the archive.

Example:

```
ActiveZipperPro1.SourceFile = "c:\myarchive.zip"
Label1.Caption = "Number of files: " & ActiveZipperPro1.zMaxFiles
```

List of Error Codes

Error #: **7250** "No files specified to compress"

Error #: **7300** "Destination not specified"

Error #: **7350** "Fatal error, archive not compatible"

Error #: **7400** "No archive in property: SourceFile"

Error #: **7450** "Compression level must be between 256 and 3065"

Error #: **7500** "No file specified for property: DecompressedFile"

Error #: **7550** "No Archive specified to Decompress"

Error #: **7600** "Cannot decompress a protected archive without a valid password"

Error #: **7650** "Cannot compress/decompress when property: StopOperation = True"

Error #: **7700** "Warning! CRC check failed"

Error #: **7750** "Cannot compress and empty file!"

Error #: **7800** "You did not specify a correct floppy disk type"

Error #: **7850** "No password specified"

Error #: **7900** "Bad header file"

Error #: **7950** "Number of disks is 0"

Error #: **8000** "Archive name not found, file may be corrupt"

Error #: **8050** "Index is greater than the number of files in ZIP archive"

Error #: **8100** "No peices of root found to assemble"

Error #: **8150** "Errors during assembly of files"

How to: Example 1

Compress all of the files in one directory using the File List Box

To run this example you need One File List Box, a Command Button, Text Box, and an ActiveZipper Pro control on a Form. This example does not use PKzip but can be adapted to do so very easily.

```
Private Sub Command1_Click()
Dim Files As String
For i = 0 To flbAdd.ListCount - 1
If flbAdd.Selected(i) Then
'Don't add the "\" to the Root Drive path. i.e. "D:\" would look like "D:\\"
If Right(flbAdd.Path, 1) <> "\" Then
Files = flbAdd.Path & "\" & flbAdd.List(i) & "," & Files
Else
Files = flbAdd.Path & flbAdd.List(i) & "," & Files
End If
End If
Next

If Len(Files) > 1 Then
Files = Left(Files, Len(Files) - 1)
ActiveZipperPro1.SourceFile = Files
ActiveZipperPro1.Password = txtPass.Text 'If you have a password or not
End If

If Files <> "" Then ActiveZipperPro1.Compress
End Sub
```

How to: Example 2

Unzip selected files in a List Box

To run this example you need One List Box with Multiselect set to 1 (Simple), a Command Button, and an ActiveZipper Pro control on a Form.

```
Private Sub Form_Load()
Dim i As Integer, ret as string
ActiveZipperPro1.SourceFile = "c:\mmtimer.zip"
ret = ActiveZipperPro1.zMaxFiles
For i = 0 To ActiveZipperPro1.zMaxFiles
List1.AddItem ActiveZipperPro1.zFileName(i)
Next i
End Sub

Private Sub Command1_Click()
Dim x As Integer, Files As String
For x = 0 To List1.ListCount - 1
If List1.Selected(x) = True Then
Files = Files & List1.List(x) & ","
End If
Next x
If Len(Files) <> 0 Then
Files = Left(Files, Len(Files) - 1)
End If
ActiveZipperPro1.OutPutFile = Files
ActiveZipperPro1.UnZipIt True
End Sub
```

Ordering

If you are using the shareware version of the control please register it and the 6 file limitation of PKzip/PKunzip will be removed. I only ask you to pay $75 after 30 days to register this component to recieve the full version and updates.

Discounts are available for bulk orders. Please email
sk@iessoft.com for the current price quote.

Order it online! (Secure Server)
You will need a valid credit card. The order will be emailed to you within 72 hours.

You can also order the source code to the control! (Secure Server)

If you prefer to do it the old fashioned way:

Please send a $75 cash/check payable to:
Shafayat Kamal
99 Anderson Avenue
Wallington, NJ
07057
USA

Please include the quantity, email address, your current address, and phone number.

SplitArchive Method

See Also: DiskSize

Syntax:

object.**SplitArchive**(*ArcName*)

Description:

Splits files according to the DiskSize property.

ArcName - A string that contains the the file to be split. *ArcName* can be any file and not only .az or .zip files. The split files will reside in the directory where the original file is located.

After the files have been split you can call the AssembleFiles method to create the original file.

Example:

```
ActiveZipperPro1.SplitArchive "c:\myarchive.az"
```