

Important : This OCX was developed to work in Win 95 or Win 98 , and Not in Winnt

## *Maquisistem LongFileNames Zip Compress OCX 2.2 - MEP*

What is new in version 2.2 : Some bugs fixed , and now we are using a new method to distribute compiled applications , the name is 'Maquisistem Easy-Package - MEP' , with this new method all the files requireds to make the ocx run are linked inside the ocx .

Our homepage

<http://members.tripod.com/~Maquisistem/index.html>

OCX to create , update and extract Zip files , compatible with Win95 longfilenames , possibility to use multiples instances of the control at same time without conflicts ( this was a bug in older versions of this control ) .

This ocx was compiled with Vb5 and service pack 3 , developed to work very well only in VB and VC++ ( you can try to use it in Delphi and another development tools , but I will not provide support to languages different of VB or VC++ ) , and this our first project using the MEP method of distribution .

[Features](#)

[Properties , Methods and Events](#)

[How to install and use](#)

[Source code](#)

[FAQ](#)

[Know problems](#)

[About the developer](#)

[Bug fixeds](#)

[Registering](#)

# Features

This control is using the Freeware [INFOZIP](#) compression scheme , to create update and extract Zip files , the files generateds are 100 percent compatibles with pkzip and winzip , and with support to win95 longfilenames and long directories names also

The time spendend to compress is the same or less than winzip

You can update , delete and add files to a Zip file

You can use multiples instances of the control at same time , and any instance running cannot affect another instance running at same time , then you can compress zip files with one instance and decompress zip files with another instance without problems.

You can compress files using wildcards , and can store the long directories names inside the zip file

## *Properties ,Methods and Events*

Property MyInstance As Long  
Member of MaqLNFZip.MaqZip  
the instance of the control

Sub AboutBox()  
Member of MaqLNFZip.MaqZip  
to show a about box

Function ZipCompress(PathOfTheFilesToCompress As String, FilesToCompress As String,  
DestinationDir As String, DestinationFile As String, Options As String) As Boolean  
Member of MaqLNFZip.MaqZip  
function to create or update a zip file

Function ZipUncompress(FileToUncompress As String, DestinationDir As String, Options As  
String) As Boolean  
Member of MaqLNFZip.MaqZip  
function to extract or update a zip file

Event ErrorCode(ErrorCode As Long)  
Member of MaqLNFZip.MaqZip  
raised when an error code appear , see the error codes in the help file to interpret the  
correct information retrieved by the error code

Event CreatingDir()  
Member of MaqLNFZip.MaqZip  
raised when a directory is created

Event Finished()  
Member of MaqLNFZip.MaqZip  
raised when the compression or the decompression is finished

Event Progress(Position As Long)  
Member of MaqLNFZip.MaqZip  
raised in the duration of the compression or decompression , the value range from 0 to  
100

# How to install and use

Due to the new MEP method of distribution , only the ocx is required , insert it in your package as a normal ocx , and the ocx will work fine in the user machine

This ocx was compiled with Vb5 and service pack 3 , developed to work very well only in VB and VC++ , read the dependency file to see all the files needed to make the ocx run

## Source code

I have included in the distribution of this package a sample project in VB5 , open it and look the code to see what you need to use the ocx

# FAQ

## What this MEP mean ?

MEP is the abbreviation of Maquisistem Easy-Package distribution , this new method was developed by Maquisistem Ltda to make possible to distribute VB5 applications with dependency files like dlls , help files , media files , and any thing that you want inside the file , but to the end user or the developer only the application or ocx will appear , and the requireds files will be automatically extracted from inside the application or ocx in the first utilization of it

## Can I use this MEP with my applications ?

At this moment this is being tested , and maybe in the future this will be available to anybody

## Where can I learn more about it ?

This is our link demonstrating it working  
<http://members.tripod.com/~Maquisistem/MEP.htm>

If I am a registered user and I have developed an application using this ocx , what I need to include in the distribution to make it work correctly in the end user machine ?

Due to the new MEP method of distribution , only the ocx is required , insert it in your package as a normal ocx , and the ocx will work fine in the user machine

## Do I need to register it in the registry of the end user machine ?

Yes , the ocx is different , but it work like a normal OCX , and need to be registered to be available in the end user machine

## How many instances of the ocx can be started at same time ?

How many as you want , because now , each instance is totally independent of others instances running ( this was a bug in older versions of this control )

## If I distribute the ocx with my application , then my end user can use this ocx for free ?

of course not , because he dont have the license file to use it in design mode , if you register then you will receive an installation program to insert the license file only in your machine , and your end user can execute your compiled application , but not use the ocx , and an error will occur when he try to insert the registered ocx in a form without the license file

## If your ocx is using the MEP method , then what are the files linked inside the OCX ?

The files linked inside are the INFOZIP Zip compression executables, these files are freeware , and anyone can use it without problems in freeware or comercial software , these files was developed by the [INFOZIP](#) group .

Is your ocx compatible with long filenames and folders ?

Yes , the ocx is fully compatible with win95 longfilenames and long directories names , and 100 percent compatible with pkzip 2.04g , pkzip 2.50 and winzip

Can I add files to a already created zip file

Yes , you only need to call a already existent zip file , and if the zip file contain a filename or directory with the same name , then the file will be replaced , if not , the new entry will be added to the zip file

Can I delete files in a zip file ?

Yes , to do it call the zip file with the option -d like it:

```
MaqZip1.ZipCompress "c:\mypath", "*.txt", "c:\my another path", "myzipfile.zip", "-d"
```

this command will delete any txt file inside the "myzipfile.zip" in the folder "c:\my another path"

Do I need to insert double quotation marks around long filenames or long directory names ?

No , the ocx do it to you , you only need to provide the location , and the double quotation marks was provided by the ocx , like it :

c:\my long directory

c:\my long directory\my long zip file.zip

these information will be automatically interpreted by the ocx as :

"c:\my long directory"

"c:\my long directory\my long zip file.zip"

More question ask the developer in:

[Maquisistem@geocities.com](mailto:Maquisistem@geocities.com)

homepage

<http://members.tripod.com/~Maquisistem/index.html>

MEP homepage

<http://members.tripod.com/~Maquisistem/MEP.htm>

# Know problems

I have tested it a lot , and for the moment I cannot find a know problem to insert here



# About the developer

My name is Ricardo Santos Pereira , and I am the developer of the Maquisistem Ltda .

Thank you for using my activex control , I hope you find this control usefull , i work hard to make this control very easy to use and setup.

And if you have some question , please send email to [maquisistem@netpar.com.br](mailto:maquisistem@netpar.com.br) or [maquisistem@geocities.com](mailto:maquisistem@geocities.com) , and I gonna aswer your question so quick as possible

And watch our home page

<http://members.tripod.com/~Maquisistem/index.html>

Email

[Maquisistem@geocities.com](mailto:Maquisistem@geocities.com)  
[Maquisistem@netpar.com.br](mailto:Maquisistem@netpar.com.br)

My address its:

country: Brazil  
State: Paraná  
city: Curitiba  
address: Clara Tedesco 2862  
fone: 55 041 376 6370  
postal code: 81670-290

thank you for your interest

## Register

The price to register this ocx , will oscillate between 50U\$ and 10U\$ , or maybe freeware

What I am selling is my work over the ocx , and not the freeware **INFOZIP** files

Homepage of the INFOZIP Group  
<http://quest.jpl.nasa.gov/Info-ZIP>

To register this ocx , visit our homepage and search for registration

This is a link to automatically connect you to our homepage  
<http://members.tripod.com/~Maquisistem/index.html>

***Maquisistem Ltda***

# Error codes

These are the error codes returned after the compression or decompression with the ErrorCode Event

- 0 normal; no errors or warnings detected.
- 2 unexpected end of zip file.
- 3 a generic error in the zipfile format was detected , processing may have completed successfully anyway; some broken zipfiles created by other archivers have simple work- arounds.
- 4 Mzip32.dll was unable to allocate memory for one or more buffers during program initialization.
- 5 a severe error in the zipfile format was detected , processing probably failed immediately.
- 7 invalid comment format
- 8 Mzip32.dll -T failed or out of memory
- 9 The user aborted Mzip32.dll prematurely
- 10 Mzip32.dll encountered an error while using a temp file
- 11 read or seek error
- 12 Mzip32.dll has nothing to do
- 13 missing or empty zip file
- 14 error writing to a file
- 15 Mzip32.dll was unable to create a file to write to
- 16 bad parameters arguments
- 18 Mzip32.dll could not open a specified file to read
- 19 Invalid path of the files to compress
- 20 Invalid input files to compress
- 21 Impossible to create the directory to insert the zip file created
- 22 Unable to find the Mzip32.dll file to create or update a zip file
- 23 Invalid input file to uncompress

- 24 Impossible to create the output directory to insert the uncompressed files
- 25 Unable to find the Munzip32.dll file to uncompress zip files
- 26 Invalid destination directory to put the compressed file
- 27 Invalid zip file to be created

# Zip and Unzip Options

These are the options to be passed to the Zip and Unzip functions using the Options argument , these arguments derived from the zip.exe and unzip.exe files by Infozip.

## ZIP OPTIONS

-A Adjust self-extracting executable archive. A self-extracting executable archive is created by prepending the SFX stub to an existing archive. The -A option tells zip to adjust the entry offsets stored in the archive to take into account this "preamble" data.

-b path  
Use the specified path for the temporary archive. For

example:

```
-b /tmp stuff *
```

will put the temporary zip archive in the directory

/tmp, copying over stuff.zip to the current directory when done. This option is only useful when updating an existing archive, and the file system containing this old archive does not have enough space to hold both old and new archives at the same time.

-c Add one-line comments for each file. File operations (adding, updating) are done first, and the user is then prompted for a one-line comment for

each file. Enter the comment followed by return, or just return for no comment.

-d Remove (delete) entries from a zip archive. For

example:

```
-d foo foo/tom/junk foo/harry/\* \*.o
```

will remove the entry foo/tom/junk, all of the files that start with foo/harry/, and all of the files that end with .o (in any path). Note that shell pathname expansion has been inhibited with

backslashes, so that zip can see the asterisks, enabling zip to match on the contents of the zip archive instead of the contents of the current

directory.

Under MSDOS, -d is case sensitive when it matches names in the zip archive. This requires that file names be entered in upper case if they were zipped by PKZIP on an MSDOS system.

- D Do not create entries in the zip archive for directories. Directory entries are created by default so that their attributes can be saved in the zip archive.
- e Encrypt the contents of the zip archive using a password which is entered on the terminal in response to a prompt (this will not be echoed; if standard error is not a tty, zip will exit with an error). The password prompt is repeated to save the user from typing errors.
- f Replace (freshen) an existing entry in the zip archive only if it has been modified more recently than the version already in the zip archive; unlike the update option (-u) this will not add files that are not already in the zip archive. For example:

-f foo

This command should be run from the same directory from which the original zip command was run, since paths stored in zip archives are always relative.

Note that the timezone environment variable TZ order for the -f, -u and -o options to work correctly.

The reasons behind this are somewhat subtle but have to do with the differences between the Unix-format file times (always in GMT) and most of the other operating systems (always local time) and the necessity to compare the two. A typical TZ value is ``MET-1MEST" (Middle European time with auto-

matic adjustment for ``summertime" or Daylight Savings Time).

- F Fix the zip archive. This option can be used if some portions of the archive are missing. It is not guaranteed to work, so you MUST make a backup of the original archive first.

When doubled as in -FF the compressed sizes given inside the damaged archive are not trusted and zip

scans for special signatures to identify the limits

between the archive members. The single -F is more reliable if the archive is not too much damaged, for example if it has only been truncated, so try this option first.

Neither option will recover archives that have been incorrectly transferred in ascii mode instead of binary. After the repair, the -t option of unzip may show that some files have a bad CRC. Such files cannot be recovered; you can remove them from the

archive using the -d option of zip.

-g Grow (append to) the specified zip archive, instead of creating a new one. If this operation fails, zip attempts to restore the archive to its original state. If the restoration fails, the archive might become corrupted. This option is ignored when there's no existing archive or when at least one archive member must be updated or deleted.

-h Display the zip help information (this also appears if zip is run with no arguments).

-i files

Include only the specified files, as in:

```
-r foo . -i \*.c
```

which will include only the files that end in .c in the current directory and its subdirectories. (Note for PKZIP users: the equivalent command is

```
pkzip -rP foo *.c
```

PKZIP does not allow recursion in directories other than the current one.) The backslash avoids the shell filename substitution, so that the name

matching is performed by zip at all directory levels.

Also possible:

```
-r foo . -i@include.lst
```

which will only include the files in the current directory and its subdirectories that match the patterns in the file include.lst.

- I Don't scan through Image files. This option is available on Acorn RISC OS only; when used, zip will not consider Image files (eg. DOS partitions

or Spark archives when SparkFS is loaded) as directories but will store them as single files.

For example, if you have SparkFS loaded, zipping a Spark archive will result in a zipfile containing a directory (and its content) while using the 'I' option will result in a zipfile containing a Spark archive. Obviously this second case will also be obtained (without the 'I' option) if SparkFS isn't

loaded.

- j Store just the name of a saved file (junk the path), and do not store directory names. By default, zip will store the full path (relative to the current path).
  - J Strip any prepended data (e.g. a SFX stub) from the archive.
  - k Attempt to convert the names and paths to conform to MSDOS, store only the MSDOS attribute (just the user write attribute from UNIX), and mark the entry
- as made under MSDOS (even though it was not); for compatibility with PKUNZIP under MSDOS which cannot handle certain names such as those with two dots.
- I Translate the Unix end-of-line character LF into the MSDOS convention CR LF. This option should not be used on binary files. This option can be used on Unix if the zip file is intended for PKUNZIP under MSDOS. If the input files already contain CR

LF, this option adds an extra CR. This ensure that unzip -a on Unix will get back an exact copy of the original file, to undo the effect of zip -I.

- II Translate the MSDOS end-of-line CR LF into Unix LF. This option should not be used on binary files.

This option can be used on MSDOS if the zip file is intended for unzip under Unix.

- L Display the zip license.

- m Move the specified files into the zip archive;

actually, this deletes the target directories/files



after making the specified zip archive. If a directory becomes empty after removal of the files, the directory is also removed. No deletions are done until zip has created the archive without error. This is useful for conserving disk space, but is potentially dangerous so it is recommended to use it in combination with -T to test the archive

before removing all input files.

#### -n suffixes

Do not attempt to compress files named with the given suffixes. Such files are simply stored (0% compression) in the output zip file, so that zip doesn't waste its time trying to compress them. The suffixes are separated by either colons or semicolons. For example:

```
-rn .Z:.zip:.tiff:.gif:.snd foo foo
```

will copy everything from foo into foo.zip, but

will store any files that end in .Z, .zip, .tiff, .gif, or .snd without trying to compress them (image and sound files often have their own specialized compression methods). By default, zip does not compress files with extensions in the list .Z:.zip:.zoo:.arc:.lzh:.arj. Such files are stored directly in the output archive. The environment variable ZIPOPT can be used to change the default

options. For example under Unix with csh:

```
setenv ZIPOPT "-n .gif:.zip"
```

To attempt compression on all files, use:

```
zip -n : foo
```

The maximum compression option -9 also attempts compression on all files regardless of extension.

On Acorn RISC OS systems the suffixes are actually filetypes (3 hex digit format). By default, zip does not compress files with filetypes in the list

DDC:D96:68E (i.e. Archives, CFS files and PackDir files).

-N Save Amiga filenotes as zipfile comments. They can be restored by using the -N option of unzip. This

option is available on the Amiga only. If -c is

used also, you are prompted for comments only for those files that do not have filenotes.

- o Set the "last modified" time of the zip archive to the latest (oldest) "last modified" time found

among the entries in the zip archive. This can be used without any other operations, if desired. For example:

- o foo

will change the last modified time of foo.zip to the latest time of the entries in foo.zip.

- P password

use password to encrypt zipfile entries (if any). THIS IS INSECURE! Many multi-user operating systems provide ways for any user to see the current command line of any other user; even on stand-alone

systems there is always the threat of over-the-shoulder peeking. Storing the plaintext password as part of a command line in an automated script is even worse. Whenever possible, use the non-echoing, interactive prompt to enter passwords. (And where security is truly important, use strong encryption such as Pretty Good Privacy instead of the relatively weak encryption provided by standard

zipfile utilities.)

- q Quiet mode; eliminate informational messages and comment prompts. (Useful, for example, in shell scripts and background tasks).
- r Travel the directory structure recursively; for example:

-r foo foo

In this case, all the files and directories in foo are saved in a zip archive named foo.zip, including files with names starting with ".", since the

recursion does not use the shell's file-name substitution mechanism. If you wish to include only a specific subset of the files in directory foo and its subdirectories, use the -i option to specify the pattern of files to be included. You should not use -r with the name ".\*", since that matches "." which will attempt to zip up the parent directory (probably not what was intended).

-R Travel the directory structure recursively starting at the current directory; for example:

```
-R foo *.c
```

In this case, all the files matching \*.c in the tree starting at the current directory are stored into a zip archive named foo.zip. Note for PKZIP users: the equivalent command is

```
pkzip -rP foo *.c
```

-S Include system and hidden files. This option is effective on some systems only; it is ignored on

Unix.

-t mmddyyyy

Do not operate on files modified prior to the specified date, where mm is the month (0-12), dd is the day of the month (1-31), and yyyy is the year. For example:

```
-rt 12071991 infamy foo
```

will add all the files in foo and its subdirectories that were last modified on or after 7 December 1991, to the zip archive infamy.zip.

-tt mmddyyyy

Do not operate on files modified after or at the specified date, where mm is the month (0-12), dd is the day of the month (1-31), and yyyy is the year. For example:

```
-rtt 11301995 infamy foo
```

will add all the files in foo and its subdirectories that were last modified before the 30 November 1995, to the zip archive infamy.zip.

-T Test the integrity of the new zip file. If the check fails, the old zip file is unchanged and

(with the -m option) no input files are removed.

-u Replace (update) an existing entry in the zip archive only if it has been modified more recently than the version already in the zip archive. For example:

`zip -u stuff *`

will add any new files in the current directory, and update any files which have been modified since the zip archive `stuff.zip` was last created/modified

(note that zip will not try to pack `stuff.zip` into itself when you do this).

Note that the `-u` option with no arguments acts like the `-f` (freshen) option.

`-v` Verbose mode or print diagnostic version info.

Normally, when applied to real operations, this option enables the display of a progress indicator during compression and requests verbose diagnostic info about zipfile structure oddities.

When `-v` is the only command line argument, and `std-out` is not redirected to a file, a diagnostic screen is printed. In addition to the help screen header with program name, version, and release date, some pointers to the Info-ZIP home and distribution sites are given. Then, it shows information about the target environment (compiler type and version, OS version, compilation date and the

enabled optional features used to create the zip executable.

`-V` Save VMS file attributes. This option is available on VMS only; zip archives created with this option will generally not be usable on other systems.

`-w` Append the version number of the files to the name, including multiple versions of files. (VMS only; default: use only the most recent version of a specified file).

`-x files`

Explicitly exclude the specified files, as in:

`-r foo foo -x \*.o`

which will include the contents of `foo` in `foo.zip` while excluding all the files that end in `.o`. The backslash avoids the shell filename substitution, so that the name matching is performed by zip at all directory levels.

Also possible:

`-r foo foo -x@exclude.lst`

which will include the contents of `foo` in `foo.zip`

while excluding all the files that match the patterns in the file `exclude.lst`.

`-X` Do not save extra file attributes (Extended Attributes on OS/2, uid/gid and file times on Unix).

`-y` Store symbolic links as such in the zip archive, instead of compressing and storing the file referred to by the link (UNIX only).

`-z` Prompt for a multi-line comment for the entire zip archive. The comment is ended by a line containing just a period, or an end of file condition (^D on UNIX, ^Z on MSDOS, OS/2, and VAX/VMS). The comment can be taken from a file:

`-z foo < foowhat`

`-#` Regulate the speed of compression using the specified digit #, where -0 indicates no compression (store all files), -1 indicates the fastest compression method (less compression) and -9 indicates

the slowest compression method (optimal compression, ignores the suffix list). The default compression level is -6.

`-@` Take the list of input files from standard input. Only one filename per line.

`-$` Include the volume label for the the drive holding the first file to be compressed. If you want to include only the volume label or to force a specific drive, use the drive name as first file name,

as in:

`-$ foo a: c:bar`

This option is effective on some systems only (MSDOS and OS/2); it is ignored on Unix.

## Unzip Options

## OPTIONS

Note that, in order to support obsolescent hardware, unzip's usage screen is limited to 22 or 23 lines and should therefore be considered only a reminder of the basic unzip syntax rather than an exhaustive list of all possible flags. The exhaustive list follows:

- Z zipinfo(1L) mode. If the first option on the command line is -Z, the remaining options are taken to be zipinfo(1L) options. See the appropriate manual page for a description of these options.
- A [OS/2, Unix DLL] print extended help for the DLL's programming interface (API).
- c extract files to stdout/screen (``CRT"). This option is similar to the -p option except that the name of each file is printed as it is extracted, the -a option is allowed, and ASCII-EBCDIC conversion is automatically performed if appropriate.  
  
This option is not listed in the unzip usage screen.
- f freshen existing files, i.e., extract only those files that already exist on disk and that are newer than the disk copies. By default unzip queries before overwriting, but the -o option may be used to suppress the queries. Note that under many operating systems, the TZ (timezone) environment variable must be set correctly in order for -f and -u to work properly (under Unix the variable is usually set automatically). The reasons for this are somewhat subtle but have to do with the differences between DOS-format file times (always local time) and Unix-format times (always in GMT/UTC) and the necessity to compare the two. A typical TZ value is ``PST8PDT" (US Pacific time with automatic adjustment for Daylight Savings Time or ``summer time").
- l list archive files (short format). The names, uncompressed file sizes and modification dates and times of the specified files are printed, along with totals for all files specified. If UnZip was compiled with OS2\_EAS defined, the -l option also lists columns for the sizes of stored OS/2 extended attributes (EAs) and OS/2 access control lists (ACLs). In addition, the zipfile comment and indi-

vidual file comments (if any) are displayed. If a file was archived from a single-case file system (for example, the old MS-DOS FAT file system) and the -L option was given, the filename is converted to lowercase and is prefixed with a caret (^).

- p extract files to pipe (stdout). Nothing but the file data is sent to stdout, and the files are always extracted in binary format, just as they are stored (no conversions).
- t test archive files. This option extracts each specified file in memory and compares the CRC (cyclic redundancy check, an enhanced checksum) of the expanded file with the original file's stored CRC value.
- T [most OSes] set the timestamp on the archive(s) to that of the newest file in each one. This corresponds to zip's -go option except that it can be used on wildcard zipfiles (e.g., ``unzip -T \\*.zip") and is much faster.
- u update existing files and create new ones if needed. This option performs the same function as the -f option, extracting (with query) files that are newer than those with the same name on disk, and in addition it extracts those files that do not already exist on disk. See -f above for information on setting the timezone properly.
- v be verbose or print diagnostic version info. This option has evolved and now behaves as both an option and a modifier. As an option it has two purposes: when a zipfile is specified with no other options, -v lists archive files verbosely, adding to the basic -l info the compression method, compressed size, compression ratio and 32-bit CRC. When no zipfile is specified (that is, the complete command is simply ``unzip -v"), a diagnostic screen is printed. In addition to the normal header with release date and version, unzip lists the home Info-ZIP ftp site and where to find a list of other ftp and non-ftp sites; the target operating system for which it was compiled, as well as (possibly) the hardware on which it was compiled, the compiler and version used, and the compilation date; any special compilation options that might

affect the program's operation (see also DECRYPTION below); and any options stored in environment variables that might do the same (see ENVIRONMENT OPTIONS below). As a modifier it works in conjunc-

tion with other options (e.g., -t) to produce more verbose or debugging output; this is not yet fully implemented but will be in future releases.

-z display only the archive comment.

## MODIFIERS

-a convert text files. Ordinarily all files are extracted exactly as they are stored (as ``binary" files). The -a option causes files identified by zip as text files (those with the 't' label in zip-

info listings, rather than 'b') to be automatically extracted as such, converting line endings, end-of-file characters and the character set itself as necessary. (For example, Unix files use line feeds (LFs) for end-of-line (EOL) and have no end-of-file (EOF) marker; Macintoshes use carriage returns (CRs) for EOLs; and most PC operating systems use CR+LF for EOLs and control-Z for EOF. In addition,

IBM mainframes and the Michigan Terminal System use EBCDIC rather than the more common ASCII character set, and NT supports Unicode.) Note that zip's identification of text files is by no means

perfect; some ``text" files may actually be binary and vice versa. unzip therefore prints ``[text]" or ``[binary]" as a visual check for each file it extracts when using the -a option. The -aa option

forces all files to be extracted as text, regardless of the supposed file type.

-b [non-VMS] treat all files as binary (no text conversions). This is a shortcut for ---a.

-b [VMS] auto-convert binary files (see -a above) to fixed-length, 512-byte record format. Doubling the option (-bb) forces all files to be extracted in this format.

-B [Unix only, and only if compiled with UNIXBACKUP defined] save a backup copy of each overwritten file with a tilde appended (e.g., the old copy of ``foo" is renamed to ``foo~"). This is similar to the default behavior of emacs(1) in many loca-



tions.

- C match filenames case-insensitively. unzip's philosophy is ``you get what you ask for" (this is also responsible for the -L/-U change; see the relevant options below). Because some file systems are fully case-sensitive (notably those under the Unix operating system) and because both ZIP archives and unzip itself are portable across platforms, unzip's default behavior is to match both wildcard and literal filenames case-sensitively. That is, specifying ``makefile" on the command line will only match ``makefile" in the archive, not ``Makefile" or ``MAKEFILE" (and similarly for wildcard specifications). Since this does not correspond to the behavior of many other operating/file systems (for example, OS/2 HPFS, which preserves mixed case but is not sensitive to it), the -C option may be used to force all filename matches to be case-insensitive. In the example above, all three files would then match ``makefile" (or ``make\*", or similar). The -C option affects files in both the normal file list and the excluded-file list (xlist).
- j junk paths. The archive's directory structure is not recreated; all files are deposited in the extraction directory (by default, the current one).
- L convert to lowercase any filename originating on an uppercase-only operating system or file system.  
  
(This was unzip's default behavior in releases prior to 5.11; the new default behavior is identical to the old behavior with the -U option, which is now obsolete and will be removed in a future release.) Depending on the archiver, files archived under single-case file systems (VMS, old MS-DOS FAT, etc.) may be stored as all-uppercase names; this can be ugly or inconvenient when extracting to a case-preserving file system such as OS/2 HPFS or a case-sensitive one such as under Unix. By default unzip lists and extracts such filenames exactly as they're stored (excepting truncation, conversion of unsupported characters, etc.); this option causes the names of all files from certain systems to be converted to lowercase.

- M pipe all output through an internal pager similar to the Unix `more(1)` command. At the end of a screenful of output, `unzip` pauses with a `--More--` prompt; the next screenful may be viewed by pressing the Enter (Return) key or the space bar. `unzip` can be terminated by pressing the `q` key and, on some systems, the Enter/Return key. Unlike Unix `more(1)`, there is no forward-searching or editing capability. Also, `unzip` doesn't notice if long lines wrap at the edge of the screen, effectively resulting in the printing of two or more lines and the likelihood that some text will scroll off the top of the screen before being viewed. On some systems the number of available lines on the screen is not detected, in which case `unzip` assumes the height is 24 lines.
- n never overwrite existing files. If a file already exists, skip the extraction of that file without prompting. By default `unzip` queries before extracting any file that already exists; the user may choose to overwrite only the current file, overwrite all files, skip extraction of the current file, skip extraction of all existing files, or rename the current file.
- N [Amiga] extract file comments as Amiga filenotes. File comments are created with the `-c` option of `zip(1L)`, or with the `-N` option of the Amiga port of `zip(1L)`, which stores filenotes as comments.
- o overwrite existing files without prompting. This is a dangerous option, so use it with care. (It is often used with `-f`, however, and is the only way to overwrite directory EAs under OS/2.)
- P password use password to decrypt encrypted zipfile entries (if any). THIS IS INSECURE! Many multi-user operating systems provide ways for any user to see the current command line of any other user; even on stand-alone systems there is always the threat of over-the-shoulder peeking. Storing the plaintext password as part of a command line in an automated script is even worse. Whenever possible, use the non-echoing, interactive prompt to enter passwords.

(And where security is truly important, use strong encryption such as Pretty Good Privacy instead of the relatively weak encryption provided by standard zipfile utilities.)

- q perform operations quietly (-qq = even quieter). Ordinarily unzip prints the names of the files it's extracting or testing, the extraction methods, any file or zipfile comments that may be stored in the archive, and possibly a summary when finished with each archive. The -q[q] options suppress the printing of some or all of these messages.
- s [OS/2, NT, MS-DOS] convert spaces in filenames to underscores. Since all PC operating systems allow spaces in filenames, unzip by default extracts filenames with spaces intact (e.g., ``EA DATA. SF"). This can be awkward, however, since MS-DOS in particular does not gracefully support spaces in filenames. Conversion of spaces to underscores can eliminate the awkwardness in some cases.
- U (obsolete; to be removed in a future release) leave filenames uppercase if created under MS-DOS, VMS, etc. See -L above.
- V retain (VMS) file version numbers. VMS files can be stored with a version number, in the format file.ext;##. By default the ``;##" version numbers are stripped, but this option allows them to be retained. (On file systems that limit filenames to particularly short lengths, the version numbers may be truncated or stripped regardless of this option.)
- X [VMS, Unix, OS/2, NT] restore owner/protection info (UICs) under VMS, or user and group info (UID/GID) under Unix, or access control lists (ACLs) under certain network-enabled versions of OS/2 (Warp Server with IBM LAN Server/Requester 3.0 to 5.0; Warp Connect with IBM Peer 1.0), or security ACLs under Windows NT. In most cases this will require special system privileges, and doubling the option (-XX) under NT instructs unzip to use privileges for extraction; but under Unix, for example, a user who belongs to several groups can restore files

owned by any of those groups, as long as the user

IDs match his or her own. Note that ordinary file attributes are always restored--this option applies only to optional, extra ownership info available on some operating systems. [NT's access control lists do not appear to be especially compatible with OS/2's, so no attempt is made at cross-platform

portability of access privileges. It is not clear under what conditions this would ever be useful anyway.]

-\$ [MS-DOS, OS/2, NT] restore the volume label if the extraction medium is removable (e.g., a diskette). Doubling the option (-\$\$) allows fixed media (hard disks) to be labelled as well. By default, volume labels are ignored.

## Bugs fixeds

2.1 Bug : - the arguments of the 'Options' in the unzip function requires a space in the end to work , this was fixed in the version 2.2

demonstration

this dont work in the version 2.1 `"-o -j"` , to work you need to insert a space in the end `"-o -j "`

this was fixed in the version 2.2 , and now this option work `"-o -j"`

