Welcome!

Zip Studio 1.2 is a Zip\Unzip API for Windows 3.10. Zip Studio contains a set of DLLs, include files, demo program and sources files and this Winhelp help file. Zip Studio is PKZip 2.04g compatable and I hope this product will help you to include Zip functionalities to your favourite softwares.

Overview

Why zipping?

Zip file compression methods are actually the most popular in the PC world. Why? Certainly because it' s a very powerful tool to compress and it' s easy to use. Zipping\Unzipping is safe and fast, also, Windows programs become bigger and bigger so it' s quite compulsory to zip files to exchange them or to save costly hard disk space for example. With ZIP files, you can for example, save multiple files into one ZIP file, add comments, directories with an incredible compression rate (about, up to 80% saved space).

If you want to know more about zip files, you can download PKWare's products.

Why Zip Studio?

At the beginning, Zip was made to run under DOS (and other OS) as an utility EXE file. With Windows, maybe
you don' t like to call "external" DOS EXE file. It' s slow, not really nice and, obviously, it doesn' t gain any advantage of the Windows environment. To hide this, you can find Windows sharewares which use a Windows interface to call a Dos Zip utility. If you' he already used these, maybe you think there are some natural limitations for these products: Batch configuration is not really easy, Dialog between interface and program is a few limited and when you zip you need to buy 2 products: the interface and the utility software...

Forget it! Now, with Zip Studio you can Zip\Unzip without concession. There are many ways to use my product: For example, you' r not a programmer but you use a well kown Word processor or a DataBase product. This one doesn' t allow you to compress your document and these, because they are really powerful (..), need an
enormous disk space! Will you buy a new hard disk every 3 months? Sure, you won't. With Zip Studio you will integrate new Zip\Unzip functionalities to your software. So you will save hard disk space and, if you are using diskette backup, one diskette will be big enough to make a backup or, if you prefer a tape backup, you will save a precious compression time. Second case, you are a programmer: Now, with Zip Studio you are free to read and write ZIP files in your works. Don' t you think it' s a good value?

Really powerful?

Zip Studio is distributed on 3 DLLs. To Zip, 2 DLLs are required, to Unzip, just one. These libraries contain about 30 high level functions. I don' t give low level functions because I hope you will find my product is very easy to use and I don' t want to afraid anyone. There are 3 basic functions: One to zip, another to unzip and the last one to view a file in a zip. The unzip DLL also contains information functions. You can have a look at the ZDEMO example for these basic functions and for notification messages treatment. Functions perform any required operations: for example, you don' t have to check if a file exist before calling them.

Because Zip Studio is really easy to use, I also provide a BASIC include file.
As a conclusion, if you wish to create a fully integrate Zip shell, you can use Zip Studio. Zip Studio includes advanced functions to test, repare, analyse, use comments, recurse directories..and much more. You can have a look at FMIZ (File Manager Integrated Zip) which uses Zip Studio.

Features?

All zip basic operations are available with Zip Studio.
Because of the DLL format, DLLs can't allocate more than 64K. So when you compare my DLLs ZIP result you will notice that an EXE file is more powerful than my DLLs: I don' t think it' s because I'm a bad programmer, but EXE programs can allocate 1024K against my poor 64K...So EXE zipper are faster than me and produce smaller files...

But, don' t panic:

1/ When Zip Studio will be available on Windows NT (or 32), these limits will completly disappeared,
2/ When you zip\unzip with Zip Studio, you can use another sofware (with the MULTITASK mode),
3/ Generated zip file are only about 10% bigger and, for big files, it' s quite less than this.

I have checked the Zip DLLs with a big Zip file with 3000 files and I have successfully add 3000 other files to it. So I don' t think you will find memory lacks during zip process, except, if you backup all your hard disk. In such case, you need to buy a DOS zipper!

Not registered version?

With the unregistered version, you can fully test Zip Studio. You will see Shareware principles reminders when you initialise the DLLs. You cannot specify directory in which you want to zip\unzip your files: When you zip, the ZIP file will be placed in the "WINDOWS\TEST" sub directory and when you unzip, all expanded files will also be placed in this directory. Distributing programs made with Zip Studio unregistered version is forbidden!

Support?

I offer a technical support before and after you register. You can join me thanks to COMPUSERVE (the best way), or thanks to the parisian BBS "The Windows Manor", or, if you are really desperate and registered, you can write to me. If you ask for example, "How can I load a library with Visual Basic" or "Can I use your product to zip a file?" you will never receive any reply! I hope you understand that the technical support is provided only for real questions regarding my product (and only Zip Studio) and if there is no answer in this help file.

Price?

Everybody can buy this shareware: It only costs 55$- or 300FF-. When you pay, you can freely distribute programs with Zip Studio functionalities. No royalties. If you are an international customer and if you pay with a check drawn on an US Bank, you will also receive the last release on a diskette.

First feet with Zip Studio

Step 1 - Creating a Zip Studio project

Now, you certainly wish to create a sofware which calls Zip Studio functions. Call your development software, create a new project, select files and put ZDLL12A.DLL and ZDLL12B.DLL in your project path if you want to use Zip functions and do the same with UZDLL12.DLL if you want to do some unzipping.

Depending on the software (language) you use, you will link or not with "Importation Libraries". I only give LIB files for Visual C++, il you use BC++ for example, you will need to recreate these LIB files with IMPLIB or IMPLIBW. You must not make an import library for ZDLL12B.DLL. When your LIB files are correct, include them to your current project: Add ZDLL12A.LIB if you wish to ZIP and\or UZDLL12.LIB if you want to UNZIP.

If you are using a BASIC like language, you do not have to create or link LIB files.
Now your project is OK, call your main BAS, C, CPP, XXX file...

Step 2 - Declaring Zip Studio functions

Choose the file in which you want to initialise Zip Studio (generally this file is the main file of your project). If you are a C\C++ programmer, at the beginning of this file, just under the "#include <windows>", add "#include <zip.h>" and make sure that ZIP.H is in your INCLUDE specification directory. If you are a BASIC programmer, open ZIP.BAS and copy and paste interesting functions to your GLOBAL.BAS main file.

Step 3 - Calling the Zip Studio DLLs

Before you call all the functions you want, you must call ZipInit (if you want to use Zip functions) and \ or UnzipInit (if you want to use Unzip functions). You will do it at the beginning of your program. Afterwise, you can call all functions you want. When you will distribute your program, don' t forget to put required DLL in your package! That' s all. You can also use Setup Studio 2.2 (SSETUP22.ZIP) to create an installation program for your works.

 *Zip functions*

Zip functions allow you to create a zip file, to add or to delete files into this one and to put a comment into it. You can also repare this file if you think it has been damaged. All functions are placed in ZDLL12A.DLL and you must put ZDLL12A.DLL and ZDLL12B.DLL in your application path.

ZipInit	**Init the ZDLL12A.DLL.**
AddFileToZip	**Add a file to a ZIP file.**
ZipDeleteFiles	**Delete file(s) from a ZIP file.**
ZipRepare	**Recreate a ZIP file**
ZipSetComment	**Change the ZIP file comment**
ZipSetLanguage	**Fix the replace Dialog Box language**
ZipGetReplaceFlag	**Get the user choice overwrite mode**

See also...

- **Using Zip functions**
- **The first program**
- **Unzip functions**
- **Overview**

ZipInit

void WINAPI ZipInit(LPCSTR szYourName, LPCSTR szPassword);

Description

This function must be called before any call to a ZIP function. When you register, I give you the 2 required passwords. If you are not a registered user, give "TEST" for <szYourName> and "" for <szPassword>. If you are not registered, this function will display a copyright reminder.

Example

```
WinMain( ... )
   {
   ZipInit( "TEST", "");
   ...
```

Required DLLs

ZDLL12A, ZDLL12B

See also...

UnzipInit
Registered version

Uzing ZIP functions

To help you to use ZIP functions, these are common ZIP questions and answers.

String size and Case?

If not specified, all paths, files name, mask... are limited to 180 characters. If you use more than 179 characters you may have a GPF, so check the user input. All functions are not case sensitive.

When do I need to call ZipInit function?

If you want to use ZIP functions (which are in ZDLL12A.DLL and ZDLL12B.DLL) you must call, before any call to these functions, ZipInit .Thanks to this function, you will specify if you' re or not a registered user. Generally, you will call this function at the beginning of your program to avoid code surcharge and the MessageBox if you are not a registered user. Also, if you want to you use UNZIP functions too, you need to call both UnzipInit and ZipInit.

Why Zipping process seems to be MONOTASK?

When you use AddFileToZip function, you must give a valid Window HANDLE to activate the backgound ZIP processing (multitask mode). If you give NULL, backgound processing mode is cancelled.

Why the ZipDeleteFiles doesn' t work?

When you use this function, you must specify paths if they are stored in the ZIP file. Else, files won' t be deleted.

What must I do with notification messages?

Notification messages are a powerful feature of Zip Studio. All functions which really process the ZIP file, will send you such messages. When you add an handler for these messages, it will work when you call ZIP functions (not only the AddFileToZip function).

Background processing?

When you use the background processing mode, don' t forget that the user can also close your application before the ZIP process is complete. That' s why you must prevent the user from closing your application during the process.

Speed up process?

If you choose the background processing mode and if you use ZN_* notification messages, you must respond to these messages as soon as possible. When the ZIP process send a message to your main Window, it waits for the response, so if you make long treatment in your notification messages handler, the process will require up to 10x the original time! This is especially true with the ZN_ZIPPING message.

Storing directories?

In the Windows environment, we generally use full names instead of names without paths. When you choose to store paths in the ZIP file, this one will be relative to the ZIP path: For example, if the ZIP file is "C:\DIR1\MYZIP.ZIP" and the file to add, "C:\DIR1\DIR2\ MYFILE.TXT", the stored full name will be "DIR2/MYFILE.TXT"...

AddFileToZip

int WINAPI AddFileToZip(LPCSTR szZipFile, LPCSTR szMask, int OverwriteMode, BOOL bStorePath, BOOL bRecurse, HWND hParentWnd);

Description

This is the main and only one ZIP function. <szZipFile> indicates the ZIP file to use or to create. If you are not a registered user, this ZIP file will be placed in "WINDOWS\TEST" directory. Use this function to add one or more file(s) specified with <szMask> to the ZIP file. <szMask> can use jokers like * or ? to make a filter, or specify a DOS filename. If you wish to receive notification messages, or, if you want to activate the background processing mode, you must give your main Window HWND as <hParentWnd>, elsewise you will specify NULL.

<OverwriteMode> is one of these constants:

Name	*Value*	*Will*
OVERWRITE_ALWAYS	0	Always add files to ZIP file (default).
OVERWRITE_NEVER	1	Never overwrite files in ZIP file.
OVERWRITE_PROMPT	2	Ask the user if he wish or not to overwrite the file.
OVERWRITE_UPDATE	3	Overwrite if ZIP file version is older

<bStorePath> is one of these constants:

Name	*Value*	*Will*
STOREPATH_NO	0	Keep the file relative path.
STOREPATH_YES	1	Don' t keep the path.

<bRecurse> is one of these constants:

Name	*Value*	*Will*
RECURSE_NO	0	Add files in sub directories.
RECURSE_YES	1	Do not recurse sub directories.

This function return one of the following value:

Name	*Value*	*Error level*
ZERROR_OK	0	OK
ZERROR_WARNING	1	Just a warning (not an error).
ZERROR_DESTFILE	2	Destfile (*.ZIP) error.
ZERROR_INTERNAL	3	Internal error.
ZERROR_FORMAT	4	Not a ZIP file.
ZERROR_NOMEM	5	Not enough memory to zip.
ZERROR_NOFILE	6	Nothing to do (can be just a warning).
ZERROR_NODLL	7	Can' t find ZDLL12B.DLL.
ZERROR_COMMENT	8	<not available here>

If you give a valid <hParentWnd>, you will also receive these messages:

Name	*Value*	*Means*	*WParam*	*LParam*
ZN_ZIPPING	WM_USER + 46	Zipping a file.	Compressed size in Ko.	Current file name
ZN_FILEZIPPED	WM_USER + 45	File zipped	Compression rate.	Current file name
ZN_WRITING	WM_USER + 48	Replace Zip file	-	Zip file name
ZN_DELETING	WM_USER + 47	<na>	-	Current file name

ZN_REPARING	**WM_USER + 49**	**<na>**	**Step: (1)Read\ Write (2)Checking**	**(1)Current file name (2)Zip file name**
ZN_COMPUTE	**WM_USER +52**	**Count files before**	**Number of files**	**-**

Tips

1/ Using ZN_COMPUTE

It' s important to receive message because processing ZIP file before any action can take more than 30 seconds if your ZIP file contains more than 1000 files. ZN_COMPUTE notification message is sent to your <hParentWnd> before this action and tell you how many files will be treated.

2/ Make the user happy...

If you use the OVERWRITE_PROMPT mode, a dialog box will be shown to user, when necessary, and ask him to choose an overwrite mode (YES | NO | ALL | NEVER). Also, the user can change the overwrite mode. You have to read the new chosen overwrite mode with the ZipGetReplaceFlag function before you call again the main ZIP function.

3/ Bug in ZN_COMPUTE?...

I find some troubles with VC++, a CString, wsprintf, and the wParam value of ZN_COMPUTE. So if you use these, maybe you will find them too! I cant' get the right value for wParam when I use wsprintf (CString.GetBuffer(180), "%s ... %i", ..., (int)wParam);... When I use _itoa and CString Message += <ZN_COMPTE wParam value converted to a CString with _itoa>, it works well...so I 've used this method in ZDEMO program!

Example

```
OnLaunchZipProcess(...)
  {
  char szZipFile[180];
  char szMask[180];
  int OverwriteMode = OVERWRITE_PROMPT, i;

  lstrcpy( szZipFile , CurrentZipFile );
  lstrcpy( szMask , FilesToZip );
  for ( i=0 ; i < 2 ; i++ )
    {
    lstrcpy( szMask , GetNextFileToZip(i) );
    if ( AddFileToZip( szZipFile, szMask, OverwriteMode, STOREPATH_NO,
    RECURSE_NO, hWnd ) > 0)
      {
      MessageBox( hWnd, "Error!","Zip Studio", MB_OK );
      return 0L;
      }
    OverwriteMode = ZipGetReplaceFlag(); // change the Overwritemode
    }
  }...
```

Required DLLs

ZDLL12A, ZDLL12B

See also...

ZipGetReplaceFlag
ZipDeleteFiles

ZipSetLanguage
ZipSetComment
IsFileUnzipable

ZipDeleteFiles

int WINAPI ZipDeleteFiles(LPCSTR szZipFile, LPCSTR szFiles, HWND hParentWnd);

Description

This function deletes files in a ZIP. <szZipFile> indicate the ZIP file, <szFiles> is the file name or the filter (with * and ?) for files you wish to delete, and <hParentWnd> is the notification messages receiver (your main Window) or NULL if you wish to cancel the background processing.

Returned value is one of the following:

Name	*Value*	*Error level*
ZERROR_OK	0	OK
ZERROR_WARNING	1	Just a warning (not an error).
ZERROR_DESTFILE	2	Destfile (*.ZIP) error.
ZERROR_INTERNAL	3	Internal error.
ZERROR_FORMAT	4	Not a ZIP file.
ZERROR_NOMEM	5	Not enough memory to zip.
ZERROR_NOFILE	6	Nothing to do (can be just a warning).
ZERROR_NODLL	7	Can' t find ZDLL12B.DLL.
ZERROR_COMMENT	8	<not available here>

Remark

If a ZIP file includes directories specification, you must also indicate a path in <szFiles>, for example, if ZIP files are "DIR1/FILE1.BAK", <szFiles> will be "DIR1\FILE1.BAK" or "DIR1*.BAK"

Example

```
if ( ZipDeleteFiles( "MYZIP.ZIP", "*.BAK", hWnd )>0 )
  MessageBox( hWnd, "Can' t delete *.BAK in MYZIP.ZIP","",MB_OK );
```

Required DLLs

ZDLL12A, ZDLL12B

See also...

AddFileToZip
ZipSetLanguage
ZipSetComment

ZipRepare

int WINAPI ZipRepare(LPCSTR szZipFile, HWND hParentWnd);

Description

The ZipRepare function try to recreate a damaged ZIP file. You can also use IsThisFileAZipFile function to test the ZIP file before calling this function. <szZipFile> is the ZIP file name, <hParentWnd> is the notification messages receiver Window (your main Window) or NULL to cancel the background processing.

This function returns one of the following constants:

Name	*Value*	*Error level*
ZERROR_OK	0	OK
ZERROR_WARNING	1	Just a warning (not an error).
ZERROR_DESTFILE	2	Destfile (*.ZIP) error.
ZERROR_INTERNAL	3	Internal error.
ZERROR_FORMAT	4	Not a ZIP file.
ZERROR_NOMEM	5	Not enough memory to zip.
ZERROR_NOFILE	6	Nothing to do (can be just a warning).
ZERROR_NODLL	7	Can' t find ZDLL12B.DLL.
ZERROR_COMMENT	8	<not available here>

Example

```
if ( ZipRepare( "MYZIP.ZIP", hWnd) > 0 )
  MessageBox( hWnd, "MYZIP.ZIP is a desperate case!"
```

Required DLLs

ZDLL12A, ZDLL12B

See also...

IsThisFileAZipFile

ZipSetComment

int WINAPI ZipSetComment(LPCSTR szZipFile, LPCSTR szComment, HWND hParentWnd);

Description

This function adds, changes or, deletes the ZIP comment. ZIP comment is global to all files. A ZIP file can contain a comment up to 64 Ko, anyway, this function doesn' t accept comment bigger than 4 Ko. <szZipFile> is the ZIP file name, <szComment> the new comment or "" if you wish to delete the comment and, <hParentWnd> is the notification messages receiver Window (your main Window) or NULL to cancel the background processing. When you use this function, all the ZIP file is computed, so it' s a good idea to use the ZN_COMPUTE notification message.

This function returns one of the following constants:

Name	*Value*	*Error level*
ZERROR_OK	0	OK
ZERROR_WARNING	1	Just a warning (not an error).
ZERROR_DESTFILE	2	Destfile (*.ZIP) error.
ZERROR_INTERNAL	3	Internal error.
ZERROR_FORMAT	4	Not a ZIP file.
ZERROR_NOMEM	5	Not enough memory to zip.
ZERROR_NOFILE	6	Nothing to do (can be just a warning).
ZERROR_NODLL	7	Can' t find ZDLL12B.DLL.
ZERROR_COMMENT	8	Comment is too big or NULL.

Example

```
char* szComment = "My favourite comment...";
if ( ZipSetComment( "MYZIP.ZIP", szComment, hWnd )>0 )
  MessageBox( hWnd, "No comment set","Zip Studio", MB_OK );
```

Required DLLs

ZDLL12A, ZDLL12B

See also...

GetZipCommentLength
GetZipComment

ZipSetLanguage

BOOL WINAPI ZipSetLanguage(UINT cZipLanguage);

Description

Use this function if you wish to have the FRENCH language interface for the replace dialog box. Default is set to English language (LANGUAGE_ENGLISH). Generally, you will call this function when you init the ZIP DLLs.This function returns TRUE if language is successfully set.

<cZipLanguage> is one of the following constants:

Name	*Value*	*Error level*
LANGUAGE_FRENCH	**0**	**French mode.**
LANGUAGE_ENGLISH	**1**	**English mode.**

Example

```
WinMain( ... )
   {
   ZipInit( "TEST", "");
   ZipSetLanguage( LANGUAGE_FRENCH );
   ...
```

Required DLLs

ZDLL12A, ZDLL12B

See also...

ZipInit
Using Zip functions

ZipGetReplaceFlag

int WINAPI ZipGetReplaceFlag(void);

Description

When you choose the OVERWRITE_PROMPT mode, you need to give the user choice overwrite mode when you will call AddFileToZip again. This function returns the TRUE overwrite mode which is:

Name	*Value*	*Will*
OVERWRITE_ALWAYS	**0**	**Always add files to ZIP file (default).**
OVERWRITE_NEVER	**1**	**Never overwrite files in ZIP file.**
OVERWRITE_PROMPT	**2**	**Ask the user if he wish or not overwrite the file.**
OVERWRITE_UPDATE	**3**	**Overwrite if ZIP file version is older**

Example

```
if ( AddFileToZip( szZipFile, szMask, OverwriteMode, STOREPATH_NO,
RECURSE_NO, hWnd ) > 0)
   {
   MessageBox( hWnd, "Error!","Zip Studio", MB_OK );
   return 0L;
   }
OverwriteMode = ZipGetReplaceFlag(); // change the Overwritemode
}
```

Required DLLs

ZDLL12A, ZDLL12B

See also...

AddFileToZip

Unzip functions

Unzip functions allow you to get informations or to unzip a zip file. You can also view a file in a ZIP file and, use general informations functions. All functions are placed in UZDLL12.DLL and you must put UZDLL12.DLL in your application path.

Initialisation

UnzipInit	Unzip initialisation function.

Zip informations functions

CountFileInZip	Returns the number of files in a ZIP.
GetFileNameFromZIP	Returns a filename of a file in a ZIP.
GetZFileOriginalSize	Returns the original size of a file in a ZIP.
GetZFileCompressedSize	Returns the compressed size of a file in a ZIP.
GetZFileDate	Returns the date of a file in a ZIP.
GetZFileTime	Returns the time of a file in a ZIP.
GetZCompressMethod	Returns the compression method for a file in a ZIP.
IsThisFileAZipFile	Tests a ZIP file.
IsFileUnzipable	TRUE if a filename of a ZIP is a DOS filename.
GetZipCommentLength	Returns the size of the ZIP comment.
GetZipComment	Retrieve the ZIP comment.

General informations functions

bDoesFileExist	Returns TRUE if a file exists.
GetShortFileName	Returns the file name without path.
GetPathFromFileName	Returns the path of a file name.
IsFileNameInFilter	TRUE if filename is in filter.

Extraction functions

ExtractZipFiles	Unzip a ZIP file.
GetQueryFlag	Returns the user's choice overwrite mode.

View function

ViewFileFromZip	View file(s) in a ZIP file.

Interface setting functions

UnzipSetMsgBoxTitle	Sets the replace dialog box title.
UnzipSetLanguage	Sets the replace dialog box language.
UnzipSetReceivingWindow	Sets the notification messages receiver Window.
UnzipSetBackgroundMode	En\Disable the background processing mode.

Required DLL

UZDLL12

See also...

- Using Unzip functions
- Zip functions
- Overview

Using Unzip functions

String size and Case?

If not specified, all paths, files name, mask... are limited to 180 characters. If you use more than 179 characters you may have a GPF, so check the user input. All functions are not case sensitive.

Must I use IsFileUnzipable before unzipping?

It' s not really necessary because the unzipping process tries to convert all names in DOS file name. Anyway, if you wish to be sure that all files with an UNIX like file name to be correctly unzipped, call this function. You can also use the IsFileUnzipable when you call a zip function in order to be sure the user choice is a valid DOS file name.

Background processing?

When you use the background processing mode, don' t forget that the user can also close your application before the UNZIP process is complete. That' s why you must prevent the user from closing your application during the process.

No ZN_COMPUTE?

There is no ZN_COMPUTE notification message for UNZIP functions. You can guess how much time process will require using the CountFileInZip function. Anyway, when you UNZIP, no extra time is required before the process.

Speed up process?

If you choose the background processing mode and if you use ZN_* notification messages, you must respond to these messages as soon as possible. When the UNZIP process send a message to your main Window, it waits for the response, so if you make long treatment in your notification messages handler, the process will require up to 10x the original time! This is especially true with the ZN_EXPANDING message.

UnzipInit

void WINAPI UnzipInit(LPCSTR szYourName, LPCSTR szPassword);

Description

Call this function before any call to an UNZIP function. When you register, I give you these two required passwords. If you are not a registered user give "TEST" for <szYourName> and "" for <szPassword>. Unregistered version will display a copyright reminder and when you will unzip files, they will be placed in the "WINDOWS\TEST" directory. Generally, you will call this function at the beginning of your program.

Example

```
WinMain( ... )
   {
   UnzipInit( "TEST", "");
   UnzipSetLanguage( LANGUAGE_FRENCH );
   ...
```

Required DLL

UZDLL12

See also...

ZipInit
Registered version

CountFileInZip

int WINAPI CountFileInZip(LPCSTR szFileName);

Description

This function returns the number of files in a ZIP file or 0. <szFileName> specify the ZIP file name. If this function fails, -1 is returned.

Example

```
if ( CountFileInZip("MYZIP.ZIP") < 1)
  MessageBox( hWnd, "MYZIP is empty!","",MB_OK);
```

Required DLL

UZDLL12

See also...

GetFileNameFromZIP
IsThisFileAZipFile
IsFileUnzipable

GetFileNameFromZIP

LPSTR WINAPI GetFileNameFromZIP(LPCSTR szFileName, UINT iFileNumber);

Description

This function returns the file name of a file in the ZIP file at position <iFileNumber>. <szFileName> is the ZIP file name and <iFileNumber> is the position (beginning at 1) of the file you want to retrieve the name. Returned name also includes relative path if available.

Example

```
char FirstFileInZip[180];
if ( CountFileInZip("MYZIP.ZIP") > 0 )
  lstrcpy( FirstFileInZip, GetFileNameFromZip("MYZIP.ZIP",1));
```

Required DLL

UZDLL12

See also...

CountFileInZip
IsThisFileAZipFile
IsFileUnzipable

GetZFileOriginalSize

long WINAPI GetZFileOriginalSize(LPCSTR szZIPFileName, LPCSTR szFileName);

Description

Returns the original length of a file in a ZIP file. <szZIPFileName> is the ZIP file name and <szFileName> is the fully qualified name of the file you wish to get original size. The returned value is original file size in octets, or 0L if function fails.

Example

```
char FirstFileInZip[180];
char Message[300];
long FirstFileOriginalSize = 0L;
if ( CountFileInZip("MYZIP.ZIP") > 0 )
   {
   lstrcpy( FirstFileInZip, GetFileNameFromZip("MYZIP.ZIP",1));
   FirstFileOriginalSize = GetZFileOriginalSize("MYZIP.ZIP",FirstFileInZip);
   wsprintf( Message, "%s original size is:%ld.", FirstFileInZip,
   FirstFileOriginalSize);
   MessageBox( hWnd, Message, "", MB_OK );
   }
```

Required DLL

UZDLL12

See also...

CountFileInZip
IsThisFileAZipFile
IsFileUnzipable
GetZFileCompressedSize

GetZFileCompressedSize

long WINAPI GetZFileCompressedSize(LPCSTR szZIPFileName, LPCSTR szFileName);

Description

Returns the compressed length of a file in a ZIP file. <szZIPFileName> is the ZIP file name and <szFileName> is the fully qualified name of the file you wish to get compressed size. The returned value is compressed file size in octets, or 0L if function fails.

Example

```
char FirstFileInZip[180];
char Message[300];
long FirstFileCompressedSize = 0L;
if ( CountFileInZip("MYZIP.ZIP") > 0 )
   {
   lstrcpy( FirstFileInZip, GetFileNameFromZip("MYZIP.ZIP",1));
   FirstFileCompressedSize =
   GetZFileCompressedSize("MYZIP.ZIP",FirstFileInZip);
   wsprintf( Message, "%s compressed size is:%ld.", FirstFileInZip,
   FirstFileCompressedSize);
   MessageBox( hWnd, Message, "", MB_OK );
   }
```

Required DLL

UZDLL12

See also...

CountFileInZip
IsThisFileAZipFile
IsFileUnzipable
GetZFileOriginalSize

GetZFileDate

LPSTR WINAPI GetZFileDate(LPCSTR szZIPFileName, LPCSTR szFileName);

Description

Returns the last modification date for a file in the specified ZIP file. This value is returned as a string formatted depending on the current Windows configuration (specified in WIN.INI, "intl" section, sShortDate value). Items are always separated with the / character like 00/00/00. If function fails returned value is "" or "00/00/00" if no date is available for this file. <szZIPFileName> is the ZIP file name, <szFileName> is the fully qualified file name you want to get informations about.

Example

```
char FirstFileInZip[180];
char Message[300];
char LastDate[15];
if ( CountFileInZip("MYZIP.ZIP") > 0 )
   {
   lstrcpy( FirstFileInZip, GetFileNameFromZip("MYZIP.ZIP",1));
   lstrcpy( LastDate, GetZFileDate("MYZIP.ZIP",FirstFileInZip));
   wsprintf( Message, "%s last modification date is:%s.", FirstFileInZip,
   LastDate);
   MessageBox( hWnd, Message, "", MB_OK );
   }
```

Required DLL

UZDLL12

See also...

CountFileInZip
IsThisFileAZipFile
IsFileUnzipable
GetZFileTime

GetZFileTime

LPSTR WINAPI GetZFileTime(LPCSTR szZIPFileName, LPCSTR szFileName);

Description

Returns the last modification time for a file in the specified ZIP file. This value is returned as a string formatted depending of the current Windows configuration (specified in WIN.INI, "intl" section, iTime value). Items are always separated with the : character like 00:00 or 00:00am or 00:00pm. If function fails returned value is "" or "00:00" if no time is available for this file. <szZIPFileName> is the ZIP file name, <szFileName> is the fully qualified file name you want to get informations about.

Example

```
char FirstFileInZip[180];
char Message[300];
char LastTime[15];
if ( CountFileInZip("MYZIP.ZIP") > 0 )
   {
   lstrcpy( FirstFileInZip, GetFileNameFromZip("MYZIP.ZIP",1));
   lstrcpy( LastTime, GetZFileTime("MYZIP.ZIP",FirstFileInZip));
   wsprintf( Message, "%s last modification time is:%s.", FirstFileInZip,
   LastTime);
   MessageBox( hWnd, Message, "", MB_OK );
   }
```

Required DLL

UZDLL12

See also...

CountFileInZip
IsThisFileAZipFile
IsFileUnzipable
GetZFileDate

GetZCompressMethod

UINT WINAPI GetZCompressMethod(LPCSTR szZipFileName, LPCSTR szFileName);

Description

This function returns the compression method used for <szFileName> in ZIP file <szZipFileName>.<szFileName> is the fully qualified name of the file you want to get informations about.

Returned value is one of the following:

Name	*Value*	*Means*
ZMETHOD_STORED	0	File is stored (Zip 1.0 compatable)
ZMETHOD_SHRUNK	1	File is Shrunked
ZMETHOD_REDUCE1	2	File is reduced (1)
ZMETHOD_REDUCE2	3	File is reduced (2)
ZMETHOD_REDUCE3	4	File is reduced (3)
ZMETHOD_REDUCE4	5	File is reduced (4)
ZMETHOD_IMPLODE	6	File is imploded
ZMETHOD_TOKEN	7	File is tokened
ZMETHOD_DEFLATE	8	File is deflated (default)
ZMETHOD_UNKNOWN	9	Not compatable with Zip Studio.
ZMETHOD_ERROR	10	ZIP file damaged

Example

```
char FirstFileInZip[180];
if ( CountFileInZip("MYZIP.ZIP") > 0 )
  {
  lstrcpy( FirstFileInZip, GetFileNameFromZip("MYZIP.ZIP",1));
  if ( GetZCompressMethod("MYZIP.ZIP",FirstFileInZip)> 8)
    MessageBox( hWnd, "Can't unzip the first file!","MYZIP.ZIP", MB_OK);
  }
```

Required DLL

UZDLL12

See also...

CountFileInZip
IsThisFileAZipFile
IsFileUnzipable
GetZFileOriginalSize
GetZFileCompressedSize

IsThisFileAZipFile

BOOL WINAPI IsThisFileAZipFile(LPCSTR szFileName);

Description

This function returns TRUE if <szFileName> doesn' t seem to be damaged, FALSE in the other case. <szFileName> is the ZIP file name. If you think the ZIP file is corrupted, you can use the ZipRepare ZIP function to recreate the ZIP file.

Example

```
if ( !IsThisFileAZipFile("MYZIP.ZIP") )
   {
   if ( ZipRepare("MYZIP.ZIP", hWnd) > 0 )
     MessageBox( hWnd, "MYZIP.ZIP is a desperate case!","",MB_OK);
   }
```

Required DLL

UZDLL12

See also...

ZipRepare
bDoesFileExist

IsFileUnzipable

BOOL WINAPI IsFileUnzipable(LPCSTR szFileName);

Description

This function returns TRUE if <szFileName> is a DOS compatable file name, FALSE if not.

Example

```
char FirstFileInZip[180];
if ( CountFileInZip("MYZIP.ZIP") > 0 )
   {
   lstrcpy( FirstFileInZip, GetFileNameFromZip("MYZIP.ZIP",1));
   if ( !IsFileUnzipable(FirstFileInZip))
     MessageBox( hWnd, "Can't unzip the first file!","MYZIP.ZIP", MB_OK);
   }
```

Required DLL

UZDLL12

See also...

GetShortFileName
GetPathFromFileName
IsFileNameInFilter

GetZipCommentLength

UINT WINAPI GetZipCommentLength(LPCSTR szFileName);

Description

Use this function to retrieve the length of the ZIP global comment. The length of this one can be as big as 64 Ko. <szFileName> is the ZIP file name. This function returns the length of the global ZIP comment (not including the last void character) or 0 if there is no ZIP global comment.

Example

```
HANDLE hComment;
char* szCommentBuffer;
UINT CommentLength = 0;
if ( (CommentLength = GetZipCommentLength("MYZIP.ZIP")) > 0 )
  {
  hComment = GlobalAlloc( GHND, (long)(CommentLength+2));
  szCommentBuffer = (LPSTR)GlobalLock( hComment );
  lstrcpy( szCommentBuffer, "");
  GetZipComment( "MYZIP.ZIP", szCommentBuffer );
  if ( lstrlen( szCommentBuffer )> 0 )
    MessageBox( hWnd, szCommentBuffer, "MYZIP.ZIP Comment", MB_OK );
  else
    MessageBox( hWnd, "Can' t get it!, "MYZIP.ZIP Comment", MB_OK );
  }
```

Required DLL

UZDLL12

See also...

GetZipComment
IsThisFileAZipFile
ZipSetComment

GetZipComment

BOOL WINAPI GetZipComment(LPCSTR szFileName, LPSTR szBuffer);

Description

Use this function to read the ZIP file global comment. To use this function, call GetZipCommentLength before. <szFileName> is the ZIP file name, <szBuffer> will be filled with the ZIP comment or "" if there is no available comment. The returned value is TRUE if comment has been read, FALSE if not.

Example

```
HANDLE hComment;
char* szCommentBuffer;
UINT CommentLength = 0;
if ( (CommentLength = GetZipCommentLength("MYZIP.ZIP")) > 0 )
  {
  hComment = GlobalAlloc( GHND, (long)(CommentLength+2));
  szCommentBuffer = (LPSTR)GlobalLock( hComment );
  lstrcpy( szCommentBuffer, "");
  GetZipComment( "MYZIP.ZIP", szCommentBuffer );
  if ( lstrlen( szCommentBuffer )> 0 )
    MessageBox( hWnd, szCommentBuffer, "MYZIP.ZIP Comment", MB_OK );
  else
    MessageBox( hWnd, "Can' t get it!, "MYZIP.ZIP Comment", MB_OK );
  }
```

Required DLL

UZDLL12

See also...

GetZipCommentLength
IsThisFileAZipFile
ZipSetComment

bDoesFileExist

BOOL WINAPI bDoesFileExist(LPCSTR szFileName);

Description

This function returns TRUE if <szFileName> exists, FALSE in the other case.

Example

```
if ( !bDoesFileExist("MYZIP.ZIP"))
  MessageBox(hWnd,"MYZIP.ZIP is not in the current directory!","",MB_OK);
```

Required DLL

UZDLL12

See also...

GetShortFileName
GetPathFromFileName
IsFileNameInFilter
IsThisFileAZipFile

GetShortFileName

LPSTR WINAPI GetShortFileName(LPCSTR szFileName);

Description

This function returns the file name without directory specification for <szFileName>. If <szFileName> doesn' t specify any directory its full value will be returned. You can use jokers (* or ?) and / or \ to specify paths for <szFileName>.

Example

```
char DOSFileName[15];
char DOSPath[180];
lstrcpy( DOSFileName, GetShortFileName("C:\\ZIP\\MYZIP.ZIP"));
lstrcpy( DOSPath, GetPathFromFileName("C:\\ZIP\\MYZIP.ZIP"));
MessageBox( hWnd, DOSFileName, "Name", MB_OK );
MessageBox( hWnd, DOSPath, "Path", MB_OK );
```

Required DLL

UZDLL12

See also...

GetPathFromFileName
IsFileNameInFilter
IsThisFileAZipFile
bDoesFileExist

GetPathFromFileName

LPSTR WINAPI GetPathFromFileName(LPCSTR szFileName);

Description

This function returns the path for <szFileName>. If <szFileName> doesn' t specify any directory "" will be returned. If there is a directory specification, this one will end with "\". You can use jokers (* or ?) and / or \ to specify paths for <szFileName>. Returned path can' t be longuer than 180 characters and ends with "\".

Example

```
char DOSFileName[15];
char DOSPath[180];
lstrcpy( DOSFileName, GetShortFileName("C:\\ZIP\\MYZIP.ZIP"));
lstrcpy( DOSPath, GetPathFromFileName("C:\\ZIP\\MYZIP.ZIP"));
MessageBox( hWnd, DOSFileName, "Name", MB_OK );
MessageBox( hWnd, DOSPath, "Path", MB_OK );
```

Required DLL

UZDLL12

See also...

GetShortFileName
IsFileNameInFilter
IsThisFileAZipFile
bDoesFileExist

IsFileNameInFilter

BOOL WINAPI IsFileNameInFilter(LPCSTR szFileName, LPCSTR szMask);

Description

This function returns TRUE if <szFileName> matches <szMask>. <szMask> can be a file name or a mask using * or ?. Also you can specify multiple masks if you separate them with a space (" "): For example, "*.BAK MYFILE.TXT" will be a valid mask. Also, you can specify directories for <szFileName> or <szMask> but they will be ignored.

Example

This example will unzip, at best, only the first file of the ZIP file, if this file matches the user specified filter

```
char FirstFileInZip[180];
char UserMask[180];
...
lstrcpy( UserMask, GetUserChoosenFilter());
...
if ( CountFileInZip("MYZIP.ZIP") > 0 )
   {
   lstrcpy( FirstFileInZip, GetFileNameFromZip("MYZIP.ZIP",1));
   if ( IsFileNameInFiler( FirstFileInZip, UserMask ))
     ExtractZipFiles( "MYZIP.ZIP",FirstFileInZip,"",OVERWRITE_TRUE,CREATEDIR_F
     ALSE);
   ...
   }
```

Required DLL

UZDLL12

See also...

GetPathFromFileName
GetShortFileName
IsFileUnzipable
IsThisFileAZipFile
bDoesFileExist

ExtractZipFiles

UINT WINAPI ExtractZipFiles(LPCSTR szFileName, LPCSTR szMask, LPCSTR szDestDir, UINT bOverwrite, BOOL bCreateDir);

Description

This is the main UNZIP function. This one will extract files specified by <szMask> of ZIP file <szFileName> into <szDestDir> destination directory if specified and, will recreate directories if they are specified in the ZIP file and if you choose the CREATEDIR_TRUE mode. If you aren' t a registered user, all unzipped files will be placed in the "WINDOWS\TEST" directory.

<szFileName> is the ZIP file name, <szMask> is the file(s) to UNZIP specification and can be a file name, a mask using jokers (* or ?) or a list of file names or masks separated with a space (like "*.BAK *.TXT"). If you don' t specified any <szMask>, "*.*" will be used. You don' t have to specify any path for <szMask>. <szDestDir> is the name of the destination directory. If you don' t give a value for <szDestDir>, and, if you are a registered user, files will be unzipped in the ZIP file directory. <szDestDir> can end with "\" or not.

<bOverwrite> is the overwrite mode and can be:

Name	*Value*	*Means*
OVERWRITE_FALSE	0	Don' t UNZIP files if they already exist.
OVERWRITE_TRUE	1	Always UNZIP files.
OVERWRITE_QUERY	2	Ask the user if he wish to overwrite a file.

<bCreateDir> can be one of of the following:

Name	*Value*	*Means*
CREATEDIR_FALSE	0	Do not recreate dir if stored in the ZIP file.
CREATEDIR_TRUE	1	Recreate directories stored in the ZIP file.

This function returns one of the following value:

Name	*Value*	*Means*
ZEXTRACT_OK	0	OK.
ZEXTRACT_INTERNALERROR	1	Fails to unzip files.
ZEXTRACT_FILENOTFOUND	2	Can' t find a file.
ZEXTRACT_CORRUPTED	3	Not a ZIP file.
ZEXTRACT_EMPTY	4	Nothing to do (can be just a warning).
ZEXTRACT_ERRORINZIPFILE	5	ZIP file corrupted.
ZEXTRACT_NOMEM	6	Not enough memory to UNZIP.
ZEXTRACT_DISKFULL	8	Disk is full!
ZEXTRACT_WARNING	10	Just a warning.

If you want to receive process notification messages, you must use the UnzipSetReceingWindow to activate it. So you will receive these messages:

Name	*Value*	*Means*	*WParam*	*LParam*
ZN_OPENFILE	WM_USER + 38	Open a file.	TRUE if OK.	Current filename
ZN_EXPANDIN	WM_USER +	Unzipping a	Current fill rate.	Current

G ZN_CLOSEFILE	**39 WM_USER + 40**	**file. Closing a file.**	**(0..100). TRUE if OK.**	**filename Current filename**

If you use the OVERWRITE_QUERY mode, a dialog box will be shown to user, when necessary, and ask him to choose an overwrite mode (YES | NO | ALL | NEVER). Also, the user can change the overwrite mode. You have to read the new chosen overwrite mode with the GetQueryFlag function before you call again the main UNZIP function.

Example

This first example will unzip, at best, only the first and the last files of the ZIP file, if these files match the user specified filter.

```
char FirstFileInZip[180];
char LastFileInZip[180];
char UserMask[180];
int LastFilePos;
int OverwriteMode = OVERWRITE_QUERY;
...
lstrcpy( UserMask, GetUserChoosenFilter());
...
if ( (LastFilePos = CountFileInZip("MYZIP.ZIP")) > 0 )
  {
  lstrcpy( FirstFileInZip, GetFileNameFromZip("MYZIP.ZIP",1));
  if ( IsFileNameInFilter( FirstFileInZip, UserMask ))
    ExtractZipFile( "MYZIP.ZIP",FirstFileInZip,"",OverwriteMode,CREATEDIR_FAL
    SE);
  ...
  }
```

Now have a look at the overwrite mode before unzipping the last file

```
if ( LastFilePos>1 )
  {
  lstrcpy( LastFileInZip, GetFileNameFromZip("MYZIP.ZIP",LastFilePos));
  if ( IsFileNameInFilter( LastFileInZip, UserMask ))
    {
    OverwriteMode = GetQueryFlag();
    ExtractZipFile( "MYZIP.ZIP",LastFileInZip,"",OverwiteMode,CREATEDIR_FALSE
    );
    ...
    }
```

Anyway, we can rewrite this previous example like this, which dooesn' t call the GetQueryFlag function:

```
char FirstFileInZip[180];
char LastFileInZip[180];
char OurNewMask[360];
char UserMask[180];
int LastFilePos;
int OverwriteMode = OVERWRITE_QUERY;
...
lstrcpy( UserMask, GetUserChoosenFilter());
lstrcpy( LastFileInZip, "");
...
if ( (LastFilePos = CountFileInZip("MYZIP.ZIP")) > 0 )
  {
  lstrcpy( FirstFileInZip, GetFileNameFromZip("MYZIP.ZIP",1));
  lstrcpy( OurNewMask, FirstFileInZip);
```

```
    if ( LastFilePos > 1 )
    {
    lstrcat( OurNewMask, " ");
    lstrcpy( LastFileInZip, GetFileNameFromZip("MYZIP.ZIP",LastFilePos));
    lstrcat( OurNewMask, LastFileInZip );
    }
  if ( (IsFileNameInFilter( FirstFileInZip, UserMask )) &&
  (IsFileNameInFilter( LastFileInZip, UserMask )))
    ExtractZipFile( "MYZIP.ZIP",OurNewMask,"",OverwriteMode,CREATEDIR_FALSE);
  else if ( (IsFileNameInFilter( FirstFileInZip, UserMask )) && !
  (IsFileNameInFilter( LastFileInZip, UserMask )))
    ExtractZipFile( "MYZIP.ZIP",FirstFileInZip,"",OverwriteMode,CREATEDIR_FAL
    SE);
  else if ( !(IsFileNameInFilter( FirstFileInZip, UserMask )) &&
  (IsFileNameInFilter( LastFileInZip, UserMask )))
    ExtractZipFile( "MYZIP.ZIP",LastFileInZip,"",OverwriteMode,CREATEDIR_FALS
    E);
  ...
```

Required DLL

UZDLL12

See also...

GetQueryFlag
UnzipSetMsgBoxTitle
UnzipSetLanguage
UnzipSetReceivingWindow
UnzipSetBackgroundMode
IsFileUnzipable
IsThisFileAZipFile
IsFileNameInFilter
ZipRepare
AddFileToZip

GetQueryFlag

int WINAPI GetQueryFlag(void);

Description

This function returns the last Overwrite mode for unzipping. It' s useful when we choose the OVERWRITE_QUERY mode. Returned value is one of the following:

Name	*Value*	*Means*
OVERWRITE_FALSE	**0**	**Don' t UNZIP files if they already exist.**
OVERWRITE_TRUE	**1**	**Always UNZIP files.**
OVERWRITE_QUERY	**2**	**Ask the user if he wish to overwrite a file.**

Example

This first example will unzip, at best, only the first and the last files of the ZIP file, if these files match the user specified filter.

```
char FirstFileInZip[180];
char LastFileInZip[180];
char UserMask[180];
int LastFilePos;
int OverwriteMode = OVERWRITE_QUERY;
...
lstrcpy( UserMask, GetUserChosenFilter());
...
if ( (LastFilePos = CountFileInZip("MYZIP.ZIP")) > 0 )
  {
  lstrcpy( FirstFileInZip, GetFileNameFromZip("MYZIP.ZIP",1));
  if ( IsFileNameInFilter( FirstFileInZip, UserMask ))
    ExtractZipFile( "MYZIP.ZIP",FirstFileInZip,"",OverwriteMode,CREATEDIR_FAL
    SE);
  ...
  }
```

Now have a look at the overwrite mode before unzipping the last file

```
if ( LastFilePos>1 )
  {
  lstrcpy( LastFileInZip, GetFileNameFromZip("MYZIP.ZIP",LastFilePos));
  if ( IsFileNameInFilter( LastFileInZip, UserMask ))
    {
    OverwriteMode = GetQueryFlag();
    ExtractZipFile( "MYZIP.ZIP",LastFileInZip,"",OverwiteMode,CREATEDIR_FALSE
    );
    ...
    }
```

Required DLL

UZDLL12

See also...

ExtractZipFile

ViewFileFromZip

BOOL WINAPI ViewFileFromZip(LPCSTR szZipFile, LPCSTR szMask, BOOL bTextOnly);

Description

Sometimes you don' t want to UNZIP a file directly... For example, you' ve already got an unzipped file version in your ZIP file directory and you don' t know what version is the good one. Another example: You' ve bought a new CD-ROM and you' re in a hurry to have a look at its ZIP files. Sure, you don' t want to get bored in copying these files to your hard disk, unzipping them and finally watch them.

The best solution is to view them before you decide to unzip them or not. My view function do this task and use the ready to use File Manager data and programs associations. You can also choose to bypass these associations if you set <bTextOnly> to TRUE. In this case, every files will be displayed with the TXT associated program (NOTEPAD.EXE by default).

To call this function, give the ZIP file name for <szZipFile>, one or more filename or mask(s) using jokers (* or ?) and separated with a space (" ") to <szMask>.

This function will call the UNZIP process (so you will receive notification messages) but will create temporary files in the WINDOWS directory instead of normal files. When the file is completely filled, the EXE, PIF, COM or BAT program is called with this temporary file name (~UNZIP*.TMP) and when this program is ready, the function will destroy the temporary file and will process the next file.

Notification messages this function send, are described in ExtractZipFile.

This function returns TRUE if all files have been displayed, FALSE if one or more files can' t be displayed.

Example

This example display the first file of MYZIP.ZIP

```
char FirstFileInZip[180];
if ( CountFileInZip("MYZIP.ZIP") > 0 )
   {
   lstrcpy( FirstFileInZip, GetFileNameFromZip("MYZIP.ZIP",1));
   ViewFileFromZip( "MYZIP.ZIP", FirstFileInZip, FALSE );
   ...
   }
```

Required DLL

UZDLL12

See also...

ExtractZipFile
IsThisFileAZipFile

UnzipSetMsgBoxTitle

BOOL WINAPI UnzipSetMsgBoxTitle(LPCSTR szNewTitle);

Description

This function changes the title of the UNZIP replace dialog box (default is "Replace...").
<szNewTitle> is the new title of this dialog box and can' t be longuer than 160 characters.
This function returns TRUE if the new title is set, FALSE if not.

Example

```
WinMain( ... )
   {
   UnzipInit( "TEST", "");
   UnzipSetMsgBoxTitle("Overwrite?");
   UnzipSetLanguage( LANGUAGE_FRENCH );
...
```

Required DLL

UZDLL12

See also...

UnzipSetLanguage
UnzipSetReceivingWindow
UnzipSetBackgroundMode
UnzipInit

UnzipSetLanguage

void WINAPI UnzipSetLanguage(UINT iLanguage);

Description

This function changes the language used by UNZIP replace dialog box (default is the English language).
<iLanguage> is one of the following:

Name	*Value*	*Means*
LANGUAGE_FRENCH	**0**	**French language mode**
LANGUAGE_ENGLISH	**1**	**US language (default)**

Example

```
WinMain( ... )
   {
   UnzipInit( "TEST", "");
   UnzipSetMsgBoxTitle("Overwrite?");
   UnzipSetLanguage( LANGUAGE_FRENCH );
   ...
```

Required DLL

UZDLL12

See also...

UnzipSetMsgBoxTitle
UnzipSetReceivingWindow
UnzipSetBackgroundMode
UnzipInit

UnzipSetReceivingWindow

BOOL WINAPI UnzipSetReceivingWindow(HWND hWnd);

Description

You must use this function to specify the notification messages receiver Window (generally your main Window). These messages are sent by ExtractZipFiles and ViewFileFromZip. To cancel the notification give a NULL value for <hWnd> else give a valid HWND value for <hWnd>. This function returns TRUE if the <hWnd> value is valid (NULL or a real <hWnd>).

Notification messages are:

Name	*Value*	*Means*	*WParam*	*LParam*
ZN_OPENFILE	**WM_USER + 38**	**Open a file.**	**TRUE if OK.**	**Current filename**
ZN_EXPANDING	**WM_USER + 39**	**Unzipping a file.**	**Current fill rate. (0..100).**	**Current filename**
ZN_CLOSEFILE	**WM_USER + 40**	**Closing a file.**	**TRUE if OK.**	**Current filename**

Example

```
char FirstFileInZip[180];
if ( CountFileInZip("MYZIP.ZIP") > 0 )
   {
   lstrcpy( FirstFileInZip, GetFileNameFromZip("MYZIP.ZIP",1));
   UnzipSetReceivingWindow( hWnd );
   ViewFileFromZip( "MYZIP.ZIP", FirstFileInZip, FALSE );
   ...
```

Required DLL

UZDLL12

See also...

UnzipSetMsgBoxTitle
UnzipSetLanguage
UnzipSetBackgroundMode
UnzipInit

UnzipSetBackgroundMode

void WINAPI UnzipSetBackgroundMode(BOOL bBackGroundMode);

Description

If you call this function with <bBackGroundMode> set to TRUE, when you will an UNZIP function, you will be free to do something else with your computer during the UNZIP process. By default, <bBackGroundMode> is set to TRUE. Setting this value to FALSE will shorten the processing required time.

Warning!

When you use the background processing mode, don' t forget that the user can also close your application before the UNZIP process is complete. That' s why you must prevent the user from closing your application during the process.

Example

```
WinMain( ... )
   {
   UnzipInit( "TEST", "");
   UnzipSetLanguage( LANGUAGE_FRENCH );
   UnzipSetBackgroundMode( FALSE );
   ...
```

Required DLL

UZDLL12

See also...

UnzipSetMsgBoxTitle
UnzipSetLanguage
UnzipSetReceivingWindow
UnzipInit

Note for VENDORS!

Thank you for your interest, these are essential informations for (diskettes) vendors:

Product Name:	**Zip Studio 1.2.**
Product File Name:	**ZSTUD12.ZIP or ZSTUD12.EXE.**
Product ZIP file size:	**About 350 Ko.**
Product category:	**Windows Programming.**
Product Object:	**ZIP API for Windows 3.10.**
Product requirements:	**DOS 3.3, Windows 3.10, Windows 3.1 Development software, VGA card...or better.**
Product Price:	**55$- or 300Fr-**
Author:	**Denis CHEVRON, 102 rue de Maubeuge, 75010 PARIS FRANCE.**
Author COMPUSERVE ID:	**100333,27 (mail checked every day).**

Product Short Description:
Zip Studio 1.2 is a Windows 3.10 ZIP API. It allows programmers to add ZIP\UNZIP functionalities to their works. Libraries (DLLs) are PKZip 2.04g compatable, easy to use and, header files for C and BASIC are provided.

Product Long Description:
Zip Studio 1.2 is a Windows 3.10 ZIP API. It allows programmers or macro users, to add ZIP\UNZIP functionalities to their works. About 30 high level functions are distributed on 3 DLLs. All modules are PKZip 2.04g compatable. Functions are easy to use. ZIP and UNZIP run in background. This tool also provides many ZIP Informations functions and an incredible View function. Includes files are provided for C\C++ and BASIC like languages. Anyway, it' s quite possible to add ZIP functions to a word processor if this one accepts DLLs calls. Zip Studio contains a Setup program, an easy to use help file and and a VC++ example with sources. The not registered version puts files in the "WINDOWS\TEST" directory and registration costs no more than 55$. No royalties.

ZDEMO program

Description

ZDEMO program is a basic sample program using Zip Studio and written with VC++\MFC. This program allows you to select a ZIP file and to ZIP, UNZIP or VIEW files from it. Sources files are also provided for MFC programmers.
This program demonstrates the use of notification messages treatment. Anyway it' s not a powerful program and there are not many checkings: I just give this one to let you try Zip Studio and have a look at functions calls. ZDEMO is not fast: this sample works a lot with ZN messages to do something on the main Window. Doing this slow down the process, when you will write an application with Zip Studio, generally you just will display a short rate indicator in your status bar to save process time.. Also, you can find many examples along this help file.

How to use it?

First of all select a ZIP file using the FILE|SELECT FILE command. If you think you will call the UNZIP function, you must specify a valid ZIP file, else, if you wish to create a ZIP file, type a new ZIP file name in the OPEN dialog box. Afterwise, you can select FILE|ZIP to zip files to this file or FILE|UNZIP to have a look or unzip files from it.

If you call the FILE|ZIP command, you can specify source path and mask for files you wish to add to the ZIP file. If you call the FILE|UNZIP command, you can specify the mask for files you wish to unzip, this mask will be used for the VIEW command if you choose this one. If the ZIP file contains a global comment, the COMMENT button is enable. Just click on it to have a look at this comment. When you are ready click on OK to launch the process or CANCEL to abort. As this sample use the unregistered version of Zip Studio, the ZIP file will be placed in the "WINDOWS\TEST" if you choose the ZIP command, or, in case you choose the UNZIP command, all unzipped files will be placed in this directory.

Flashing view...

The ZDEMO main Window is a few flashing... When you launch a process (ZIP, UNZIP or VIEW), 3 TEXT controls will be updated when notification messages come. This treatment is not really nice, but it' s just a demo! Choose the FILE|EXIT command to quit ZDEMO.

Terms

Description

Used terms are C and SDK language type. If you do not program with these languages, perhaps you will have some difficulties to understand them. These are some explanations:

WINAPI	**FAR PASCAL (just ignore this)**
int	**Integer value.**
long	**Long integer value.**
UINT	**Signed long integer value.**
BOOL	**Integer value, 0 for FALSE, anything else for TRUE.**
char	**Character.**
char far*	**String.**
LPCSTR	**Constant String**
LPSTR	**String.**
HWND	**Window' s Handle(Integer value).**
HINSTANCE	**Application' s Handle(Integer value).**
void	**Null type (empty).**
COLORREF	**Long integer value (Colour).**
TRUE	**An integer value other than 0 (1).**
FALSE	**0**

Package

The ZIP file you' ve received contains the following files:

(uncompressed files, about):

```
    SETUP.EXE      71.000 The setup program.
    CSETUP.DLL    113.904 A Setup module.
    UZSETUP.XXX    28.334 A Setup module.
    CTL3D.DLL      14.336 A Setup module.
    SETUP.INF       3.000 A Setup module.
    SETUP.LST         500 A Setup module.
    MAIN.EXE       98.000 A Setup module.
    FINISH.EXE      5.100 A Setup Module.

    READ.ME         2.493 Basic informations file.
    ORDER.TXT       2.128 The order form.
    DESC.SDI           41 BBS file.
    FILE_ID.DIZ       214 BBS file.

    UZDLL12.DLL    57.856 The UNZIP DLL.
    ZDLL12A.DLL    12.511 The ZIP DLL part A.
    ZDLL12B.DLL    49.824 The ZIP DLL part B.
    UZDLL12.LIB     3.584 UZDLL12.LIB importation library (
                          for VC++)
    ZDLL12A.LIB     2.048 ZDLL12A.LIB importation library (
                          for VC++)
    ZIP.H           8.760 ZIP.H declaration include file
                          for C\C++
    ZIP.BAS         9.664 ZIP.BAS declaration file for
                          BASIC like languages
    ZIP.HLP        83.800 ZIP.HLP :This file

    ZDEMO.EXE     107.664 ZDEMO - ZDEMO.EXE program
    ZDEMO.MAK       4.513 ZDEMO - ZDEMO.MAK file ( VC++)
    ZDEMO.DEF         347 ZDEMO - ZDEMO.DEF file
    CZIP.CPP        5.489 ZDEMO - CZIP.CPP file
    CZIP.H            716 ZDEMO - CZIP.H file
    MAIN.CPP        1.065 ZDEMO - MAIN.CPP file
    MAIN.H            616 ZDEMO - MAIN.H file
    MAINFRM.CPP    19.680 ZDEMO - MAINFRM.CPP file
    MAINFRM.H       1.981 ZDEMO - MAINFRM.H file
    RESOURCE.H      1.443 ZDEMO - RESOURCE.H file
    STDAFX.CPP        204 ZDEMO - STDAFX.CPP file
    STDAFX.H          299 ZDEMO - STDAFX.H file
    TOOLBAR.BMP       358 ZDEMO - TOOLBAR.BMP file
    UNZIP.CPP       6.346 ZDEMO - UNZIP.CPP file
    UNZIP.H           738 ZDEMO - UNZIP.H file
    ZDEMO.CPP       3.920 ZDEMO - ZDEMO.CPP file
    ZDEMO.H           977 ZDEMO - ZDEMO.H file
    ZDEMO.ICO         766 ZDEMO - ZDEMO.ICO file
    ZDEMO.RC       10.625 ZDEMO - ZDEMO.RC file
    ZDEMO.RC2       1.538 ZDEMO - ZDEMO.RC2 file
    ZDEMODOC.CPP    1.704 ZDEMO - ZDEMODOC.CPP file
    ZDEMODOC.H        933 ZDEMO - ZDEMODOC.H file
    ZDEMOVW.CPP     2.262 ZDEMO - ZDEMOVW.CPP file
    ZDEMOVW.H       1.140 ZDEMO - ZDEMOVW.H file

    TOTAL:        742.371 44 files
```

Compressed files

You can find ZSTUD12.ZIP or ZSTUD12.EXE on COMPSERVE, BBS and thanks to diskettes

vendors. These files are about 395K.

Author

Versions

Two versions are available: The registered version and the evaluation version. The evaluation version can be freely distributed if you give the full Zip Studio package and you don' t modify any file. You can find the evaluation version:

- on COMPUSERVE, I always put my latest releases on the WUGNET forum. You can also register thanks to COMPUSERVE (go SWREG).

- Thanks to DP Tool Club,
BP 745, 59657 VILLENEUVE D' ASCQ (FAX: (16) 20.05.38.27) FRANCE.

- On the BBS "The Windows Manor" (Paris - FRANCE),
With Windows Terminal, configure your modem with 8 Bits length, No parity, 1 Bit Stop and call number (33-1)42.43.26.18 (2400 Bds) or (33-1)42.43.78.52 (14400 Bds) and follow the instructions.

With the unregistered version, you can fully test Zip Studio. You will see Shareware principles reminders when you initialise the DLLs. You cannot specify directory in which you want to put your files: When you zip, the ZIP file will be placed in the "WINDOWS\TEST" sub directory and when you unzip, all expanded files will also be placed in this directory. Distributing programs made with Zip Studio unregistered version is forbidden!

The registered version required the evaluation version because you change the old version thanks to two personal passwords. You will receive these passwords, by mail, about 3 days after you register (French users). If you have a COMPUSERVE ID, I will give you your passwords there and without delay.

Payment of the contribution

To register you can print Order form and you must send your contribution to:

Denis CHEVRON
102 rue de Maubeuge
75010 PARIS FRANCE
(Do not phone).

You can also use the compuserve registration service (go SWREG) to pay with a credit card.

When you register, you can save $$$!
For example, I distribute a ZIP shell (FMIZ for File Manager Integrated ZIP), have a look at this. If you like FMIZ and if you register it for 5 licences, you will win a licence for Zip Studio. Have a look at the FMIZ help file for more details. Also, you can save money, if you buy Zip Studio and Setup Studio together. Setup Studio is a toolkit to create professional installation programs for Windows 3.10.

Prices depend on currency you use: You can pay in French Franc currency, which is the cheapest way to pay or in another allowed currency in this list:

Currency:	ZIP STUDIO	ZIP STUDIO + SETUP STUDIO
Francs français:	300-	450-
US Dollars:	55-	85-
Deutch mark:	90-	145-
Francs belges:	1900-	3000-
Francs suisses:	79-	125-
GB Pounds:	37-	60-

Canadian Dollars:	**77-**	**120-**

If you are living in a foreign country, maybe your sharewares distributor has a foreign payment service (in USD for example) to help you to register. Such distributors can ask for a payment for this service. For informations, please ask your distributor.

Upgrades

To know about upgrades you can connect to COMPUSERVE (go WUGNET) or to the "The Windows Manor" (see higher) and read conference "Windows programming" messages. You can also read the DP Tool Club (see higher) news.

Licence

The licence is an using authorisation but not a product cession. Programs made with the unregistered version never will be distributed: Distributing programs made with Zip Studio unregistered version is forbidden! To distribute your installation programs, you must register to Denis CHEVRON. The registered version kills Copyright warning and product tiny limitations.

You can use this product on several PC but a unique copy can be used at the same time. The sell of this product without installation program is allowed for physical persons if it is an exception.

Author and technical support

You can join the author and dialogue with others Setup Studio users on COMPUSERVE (100333,27). COMPUSERVE is the best way to join me because I read my mail every day. You can also join me on the parisian BBS "The Windows Manor" (see higher). On this service you have to go to the "Windows Programming" conference. Do not hesitate to write to me your critics. You can also send a mail at my address.

I offer a technical support before and after you register. You can join me thanks to COMPUSERVE, ID 100333,27 (the best way), or thanks to the parisian BBS "The Windows Manor", or, if you are really desperate and registered, you can write to me. If you ask for example, "How can I load a library with Visual Basic" or "Can I use your product to zip a file?" you will never receive any reply! I hope you understand that the support is only provided for real questions regarding my product (and only Zip Studio) and if there is no answer in this help file.

Other products

- Setup Studio 2.2

This new release will be available during March 1994, this is a powerful toolkit to create professional installation programs for Windows 3.10. (The installation program you' ve used to install Zip Studio uses this one).

- FMIZ 1.2

This is an application made with Zip Studio. File Manager Integrated Zip allow you to use up to 50 different compression format, plus SHAR, TAR and Add a lot of new functionalities to the Windows File Manager. It' s the most easy way to manage ZIP files..Just try it! Available at the beginning of April 1994.

- VBX Studio 1.2

A collection of about 20 general VBXs. Especially designed for VC++, BC++ and VB of course. A very good value for Visual programmers.Available at the beginning of April 1994.

- HLP Studio 1.2

A DLL and a Word 2.0 for Windows macro to generate Windows 3.10 compatible help

files. I think this product will use an external reference database to speed up the help process. Available later... (this help file has been created thanks to a BETA of HLP Studio).

See also...

Overview
Order form
Shareware

Shareware?

Zip Studio is a ShareWare:

You can try it as you want before you decide or not to register. The only one way I have to sell enough licences to support this software, is to distribute it all over the world. So, if you think this product will help one of your friends, don' t hesitate to give him a copy of this product: This is your interest!

When you distribute Zip Studio, don' t change any file and give the complete package. If you UL Zip Studio on a BBS, please name it ZSTUD12.ZIP or ZSTUD12.EXE.

Obviously, if you register, you can' t distribute the registered version! Distributing programs made with Zip Studio unregistered version is forbidden!

Order form

To register you can print this help page and manually fill it, or use ORDER.TXT.
Unreadable orders will be ignored!

--- ZIP STUDIO 1.2 ORDER FORM ---

Name

```
COMPANY:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
NAME:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
FIRST NAME:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
ATTENTION TO:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
ADDRESS:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
ZIP:_ _ _ _ _ _ _ _ _ _
TOWN:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
LAND:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
COMPUSERVE ID: _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
WHERE DID YOU FIND ZIP STUDIO?:_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
```

Product(s)

```
QUANTITY:_ _ _
CURRENCY:_ _ _ _ _ _ _ _
PRODUCT(S):_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
```

PRICES:

Currency:	ZIP STUDIO	ZIP STUDIO + SETUP STUDIO
Francs français:	300-	450-
US Dollars:	55-	85-
Deutch mark:	90-	145-
Francs belges:	1900-	3000-
Francs suisses:	79-	125-
GB Pounds:	37-	60-
Canadian Dollars:	77-	120-

(Others not allowed)
These prices include shipping and VAT.
If you are an international customer and if you pay with a check drawn on an US Bank, you will also receive the latest release of chosen product(s) on a diskette. If you' ve got a COMPUSERVE ID and if you are an international customer, I will mail you your passwords before I send you your invoice, in order to avoid Postal delays.

Send this with your check to

Denis CHEVRON,
102, rue de Maubeuge
75010 PARIS FRANCE