ShareMan 1.7 <ASP> Online Help

General information

Glossary

Main menu

- More files
- Destination directory
- Activities
- Go!
- Serial number: Parameters
- Serial number: Check single
- Serial number: Start with...
- Password parameters
- Password: Test
- Password: User ID

Address file menu

- List
- Analyze fees

LHA manager menu

- Select archive
- Add
- Delete
- Extract
- Contents
- Self-extracting

Crypto menu

- Cesar method
- Shift register method
- Editor
- Encrypt
- Decrypt

Applications

What can the ShareMan do for me?

The **ShareMan** is a Windows utility for software distribution; LHA and crypto interface can also be used generally. Encryption and decryption of text and files, inserting passwords, user IDs and serial numbers with authentication digit in the final state of program development as well as compressing and copying files for distribution are the first tasks the **ShareMan** will do for you. For shipment to a registered customer the **ShareMan** offers writing serial number, customer's name and company into the program, calculating and writing a user ID and storing customer data in an address file, with each option to be turned on or off separately. All settings can be stored in parameter files for easy retrieval.

ID and crypto algorithms are included as C header file and source code to be used freely in your own programs.

Details

Typically your program with all accompanying files is in a directory of its own. From here you want to copy it to disk for a vendor or customer. In the latter case you may want to insert serial number, customer data (name, company) and user ID. Large programs have to be compressed first.

All this the **ShareMan** will do for you. Any file in a directory can be selected and compressed with LHA (copyright by Yoshi), optionally to a self-extracting archive. Source and more files can be copied to a destination directory. Serial number and customer data can be inserted into non-compressed files automatically, or a file with the user ID is written. All data are stored in an address file which features searching, editing, deleting, printing, exporting and graphical analysis.

Additional features of the **ShareMan** are helpful during program development. When using passwords they have to be encrypted and stored in an external file so they can be re-read and decrypted by the program later. Also, you may want to protect short texts, like those in the **ShareMan**'s "About..." dialog, against manipulation, or grant a user an individual ID. For all this you will find simple algorithms to be included in your own C programs. These algorithms do not claim to be unbreakable. We are not dealing with top secret stuff here, we just want to disencourage hobbyist code breakers with their disk editors or protect private data against curious friends and colleagues.

All settings of the **ShareMan** can be stored in parameter files. One of those can be included in the command line of the program manager to be loaded automatically on program start. Once the configuration of a program to be distributed is done, all you will need is the menu item Go! to create customer disks.

See also the examples of customer disk and authentication.

Address file

This submenu offers a simple address file. So far the features are limited to those interesting for software distribution.

The address file submenu contains the following items:

List

Analyze fees

LHA manager

LHA (copyright by Yoshi) is an outstanding program to compress and archive files. Unfortunately, it can be operated only via DOS command line parameters.

The LHA manager allows you to select files and features for LHA under Windows and then calls LHA with the chosen parameters. Since LHA is a DOS program the call can be delayed by several seconds. LHA must be in the same directory with the **ShareMan**.

The LHA manager submenu contains the following items:

Select archive

Add

Delete

Extract

Contents

Self-extracting

Limitation

The "/x" options for full path names are not supported because you do not enter paths directly but work your way through directory dialogs. This makes determining the path relative to the selected archive difficult. In the rare case that you need subdirectories in your archive you must use LHA via the DOS command line.

Why LHA?

From the several data compression programs of the late 80s only three have survived: PKZIP (copyright by PKWARE Inc.) for ZIP files, ARJ (copyright by Robert K. Jung) for ARJ files and LHA (copyright by Haruyasu Yoshizaki, a.k.a. Yoshi) for LZH files. In contrast to their authors' claims all three are similar in speed and reduction rate.

The only major feature LHA lacks in comparison to its competitors is multiple disk archives. They are dispensable most times, though. The big advantage of LHA is that in commercial use (which software distribution certainly is) you do not have to pay a licence fee. All you have to do is include a copyright remark. For details see "LHA.DOC".

LHA gives you compact archives and similarly compact self-extracting archives. From my experience it can certainly hold a candle to PKZIP and ARJ. And with self-extracting archives you do not have to waste a thought on compatibility.

A little theory...

Among the many cryptic names for compression programs "LHA" is the most intelligible one. In former days this program was called "LHArc", thus indicating its use for archiving. Knowing that the archive file name extension "LZH" stands for Lempel, Ziv und Huffman you have almost broken the code.

All known compression programs use the mathematic foundation these three guys have created: Lempel and Ziv substitute frequently occuring sequences with shorter ones, thus building a dictionary of abbreviatons. This dictionary is translated via a Huffman code tree into the shortest binary words possible, i.e. the most frequently occuring sequence gets the shortest code (the ASCII fixed code length of 8 bits is dropped). The latter happens dynamically, that means the frequence of occurence is recalculated during compression for each block of data.

From this explanation one can tell that compression is the better the more data in a file repeat themselves. A picture with large background areas is ideal, a text file yields a much poorer result. Good compression programs offer several variations of the above algorithm and use the most appropriate one for each file type.

The prime time of compression programs was in the late 80s. Since then, the algorithms have been brought to their maximum performance, and unless some mathematical genius comes up with a fundamentally new concept software distributors will work with PKZIP, ARJ and LHA to eternity.

Crypto

This term stands for several methods of file encryption and decryption. Optionally this can be done byte by byte (e.g. for EXE files) or line by line (for ASCII text). The difference simply is that in the former case *Carriage Returns* are treated as normal characters and coded along, but in the latter case remain unchanged, i.e. the file is considered a series of strings to be coded independently. In C terminology the first are binary files, the latter text files, with their lines no longer than 255 characters and containing only ASCII characters from 32 on (e.g. no tab!).

Regardless of the method selected encrypted files get file name extension "ENC" and decrypted files file name extension "DEC".

Crypto algorithms are provided as source and can be used by C programmers (see Using crypto algorithms in C programs). Beside on files you can also work on characters and strings, e.g. for password checks.

Available are the methods "Cesar" and "shift register". These algorithms do not claim to be unbreakable. We are not dealing with top secret stuff here, we just want to disencourage hobbyist code breakers with their disk editors or protect private data against curious friends and colleagues.

The DES method (Data Encryption Standard) is not used for four reasons: 1) It is too complex for this application. 2) Its security is yet unproven. 3) Use outside the US is undesirable. 4) It codes bit blocks into bit blocks; it was not designed to code legible characters into legible characters.

The crypto submenu contains the following items:

- Cesar method
- Shift register method
- Editor
- Encrypt
- Decrypt

Menu bars

Beside the main menu the **ShareMan** offers access to three submenus featuring an address file, an LHA manager and crypto methods. You can see a list of all menu items in the contents.

Via the command line the **ShareMan** can be restricted to one of the latter two submenus and thus works as a general utility in the specific area

Command line

The **ShareMan** allows two parameters in the command line:

If the name of a parameter file is given it will be loaded on program start. The file name extension ".SHM" is used for identification and therefore must not be omitted. To prevent you parameter file from being saved to unwanted locations you should include its full path name.

With the parameters "/LHA" and "/CRYPTO" one of these two submenus is made the main menu. Access to other menus is no more possible; instead, the **ShareMan** behaves as an independent utility, i.e. as LHA manager or crypto assistant. These parameters must always be *after* an optional parameter file name.

Analysis of the command line is rather superficial, so you should take care yourself that parameters are correct. Capital and small letters are not distinguished.

Status bar

The **ShareMan** features a status bar to display hints on program status and short help texts on menu items.

The first field of the status bar shows a hint to save the present parameter file in case it was altered.

Parameter file

A parameter file stores all settings of the **ShareMan** made by the user. This includes especially the customer data. You can store as many parameter files under MS-DOS file names as you please.

Parameter files are plain ASCII files that can be changed with any editor. Thus you can edit customer data by hand, although an edit feature is given in the address file list. A parameter file's name can be handed to the **ShareMan** via the command line. File name extension must be"SHM".

Under "Applications" the structure of parameter files is explained.

Serial number

A serial number is an arbitrary combination of up to 12 digits, capital and small letters and fixed characters. The arbitrary combination serves the purpose to make a small serial number more impressive.

Serial numbers are typically used in the way that in the program to be distributed (source file) a variable is inserted that is recognized by the **ShareMan** and replaced by the actual number when creating a customer disk (destination directory).

Digits and letters are used for calculating an authentication digit. The calculation method can be influenced by the user. This control digit can optionally be put in front of or to the end of the serial number. It allows detection of manipulations because the forger does not know the calculation method and thus has a 90% chance of a wrong guess.

Password

Passwords are inserted during program development. The user of your program selects an arbitrary one that is encrypted and stored in a file, e.g. an "INI"-File. On a later check the specified password must be compared to the stored one that has to be decrypted then.

Hint: With non-bijective crypto methods the specified password must instead be encrypted and compared to the stored and also encrypted one. A bijective method, in contrast, can always calculate the non-encrypted password from the encrypted one.

Another application is the user ID.

The **ShareMan** allows you to specify an ASCII range for your password, e.g. digits or capital letters. The encrypted password is in the same range then.

You will find more information in the example for authentication. Crypto algorithms are provided as source and can be used by C programmers (see Using crypto algorithms in C programs). They are all bijective except "IDCalculate" which is more difficult to see through.

User ID

A special form of the password is encrypting the customer's name into an individual user ID and store both in your program's "INI" file. Your program compares the pair on start, preventing the customer from modifying the name that you display in the title bar or an "About..." dialog.

You will find more information in the example for authentication. Crypto algorithms are provided as source and can be used by C programmers (see Using crypto algorithms in C programs). They are all bijective except "IDCalculate" which is more difficult to see through.

Variables

To insert serial number and customer data into the source file copied to the destination directory the **ShareMan** expects special variables.

For customer name and company the variables are "NameNameNameNameName" und "FirmFirmFirmFirmFirm", allowing up to 20 characters in each field.

For the serial number a copy of the pattern created under serial number: parameters must be included in the source file, i.e. for the types digit, capital and small letter an "0", an "A" or an "a", respectively, and not any other character of the same type. Fixed characters must be exactly the same. Remember to leave one extra character in front of or after the actual serial number for the control digit.

Principally you cannot go wrong if you leave more room in your program than the **ShareMan** will need later. Remember that after compressing the source file the variables cannot be recognized anymore, meaning that you will have to write to an uncompressed version of your program. In case your source file has been altered in the meantime the variables might not be in the positions the **ShareMan** has stored. Simply re-select your source file, the **ShareMan** will then search for variables again when you select Go!.

You will find an application of these explanations in the example of a customer disk.

Customer data

The customer data the **ShareMan** asks of you under Go! are name, company, street, postal code, city, phone, fee paid and remark, provided that you have selected the respective activity. Added automatically are date and time of disk creation, consecutive number, serial number and user ID. The consecutive number is used by the **ShareMan** to identify customer records.

Of these data serial number, name and company can be written to disk (destination directory), provided that you have selected the respective activity.

Customer data are added to the present parameter file as plain ASCII text.

New parameter file

This menu item creates a new parameter file in memory. The **ShareMan** fills it with default data. A file name is only associated the first time it it saved.

A warning is displayed in case memory contains yet unsaved data.

Open parameter file

This menu item loads a previously saved parameter file into memory. The file name extension expected is "SHM" but this is not mandatory.

A warning is displayed in case memory contains yet unsaved data.

A parameter file's name can be included in the command line to be loaded upon program start. The file name extension must be "SHM". This is useful if you always use the same settings.

Save parameter file

This menu item saves the parameter file in memory. The present file name is used. If you want to change it or have not yet given one you will have to select menu item Save Parameter File As... instead.

The file name extension expected is "SHM" but this is not mandatory.

Save parameter file as...

This menu item saves the parameter file in memory by a new name. If you want to keep the present name you will have to select menu item Save Parameter File instead.

The file name extension expected is "SHM" but this is not mandatory.

Overview

This menu item gives you a quick look at all the settings of the **ShareMan**. This is especially useful before creating a customer disk.

Selected more files are only shown by their count. Detail information is available under the respective menu item.

Exit program

This menu item exits the **ShareMan**. A warning is displayed in case memory contains yet unsaved data.

Source file

This is the program to be distributed, containing variables for serial number and customer data instead of actual data. Usually this file is on hard disk and is to be copied to disk (destination directory) where actual data are inserted.

By selecting a new source file any selection of more files is cleared. Note that menu item Select Archive automatically makes the present archive the new source file.

In case you have already copied the program to be distributed to disk source file and destination directory are the same. This is no problem for the **ShareMan** as long as the variables are present. However, remember to keep another copy with variables because those in the destination directory will be overwritten.

You will find an application of these explanations in the example of a customer disk.

More files

Beside the source file additional files can be copied to the destination directory , e.g. "README" texts. Since this is an extended selection here a double click will *not* do to select a file and leave the dialog via "OK" in one go!

Additional files must be in the same directory with the source file since their path is not stored. For reasons of limited space they are not shown in the overview but only their count. You can see the actual files by selecting this menu item again.

By selecting a new source file any selection of additional files is cleared.

You will find an application of these explanations in the example of a customer disk.

Destination directory

Here is the version of the program to be distributed that will be delivered to a customer, usually on disk. It is created from the source file the way that the **ShareMan** copies it to the destination directory and fills the contained variables for serial number and customer data with individual data. Bside the source file more files can be copied along.

In case you have already copied the program to be distributed to disk source file and destination directory are the same. This is no problem for the **ShareMan** as long as the variables are present. However, remember to keep another copy with variables because those in the destination directory will be overwritten.

You will find an application of these explanations in the example of a customer disk.

Activities

Way 1

Normally the **ShareMan** is activated when your program to be distributed is in a directory of its own, containing the variables for serial number and customer data, along with accompanying files. You select source file, more files and destination directory and then set the demanded activities here.

Usually you distribute your program on a disk that you have to format in Windows' file manager first. Then you have your source file copied onto it, containing the variables that are now filled with individual data by the **ShareMan**. Customer data are entered in a special dialog and added to the internal address file. If you only activated "get customer data" the **ShareMan** works as a simple address file.

Depending upon how much of your distribution tasks you want to do by hand you can deactivate any option. Note that you cannot write customer data to disk or calculate and write IDs if you deactivate the "get customer data" option.

Hint: In compressed files the variables cannot be recognized anymore, meaning that you will have to write to an uncompressed version of your program. Normally you do not have to compress a customer disk, though. In case your source file has been altered in the meantime the variables might not be in the positions the **ShareMan** has stored. Simply re-select your source file, the **ShareMan** will then search for variables again when you select Go!.

You will find an application of these explanations in the example of a customer disk.

Way 2

Instead of writing serial number and customer data into the program to be distributed you can also calculate a user ID and write it to disk as a file in the format of "INI" files. The further process is as above.

You should enter the customer's name in the order *first name, last name* because the **ShareMan** automatically shortens the first name to one letter to avoid problems with various lengths of first names.

You will find an application of these explanations in the example of authentication.

Go!

This menu item starts the selected activities. Once you have configured the **ShareMan** for a program to be distributed all you have to do is select this menu item. The selections of source file, more files and destination directory as well as parameters of password and serial number and the latter's number to start with are used and the existing customer data are supplemented.

You will find an application of these explanations in the example of a customer disk and the example of authentication.

Serial number: Parameters

The serial number is made up of up to 12 positions each of which can be designed as digit, capital or small letter or fixed character. The first three types are used in calculating an authentication digit, with each contribution counting positive, negative or nothing. An "A" or "a", respectively, is treated as a "0" for this calculation, etc. The control digit can be added in front or in the end.

The authentication digit's purpose is to reveal manipulation in the serial number. Use menu item Check Single to check suspicious numbers. Because the forger does not know how your control digit is calculated he has only a 10% chance to guess right.

The 12 positions aim at making a small serial number more impressive. E.g. you can turn a mere "10" into a "2-3010-1", with only the "010" being true digits and the rest being fixed characters. The **ShareMan** only uses the three digits for calculation.

Take care that the variable in the source file exactly represents the pattern designed here, i.e. it must be made up not of arbitrary letters and digits but of "A", "a" or "0". Fixed characters must be entered precisely.

You will find an application of these explanations in the example of a customer disk.

Serial number: Check single

The purpose of the serial number's authentication digit is to reveal manipulations. This menu items helps you to check suspicious numbers. Because the forger does not know your calculation formula he has only a 10% chance of a right guess. Remember to create the exact pattern under Serial Number: Parameters. The **ShareMan** checks your input to have matching types.

Serial number: Start with...

Usually your serial number should start with the lowest possible. If you have e.g. a pattern of three digits you can keep the "000" for yourself (at the same time variable) and distribute the numbers starting from "001".

If you consider this not glamourous enough or faulty writing (see menu item Go!) has left some numbers blank, this menu item gives you the chance to set the **ShareMan**'s internal counter back or forth. Your input is checked to have types according to the serial number parameters. After a correct entry you learn which consecutive number it represents. This is helpful with serial numbers consisting of letters and fixed characters. Note that the consecutive number serves the **ShareMan** to identify customer data and therefore should not be ambiguous.

You will find an application of these explanations in the example of a customer disk.

Password parameters

You can specify three keys for password encryption, each in the range of -32768 to +32767. The **ShareMan's** interface only uses two of them for a Cesar encryption, those two also limited to a range of -16 bis +16 (see also Using crypto algorithms in C programs). The keys entered can be tested on a password or be used for calculating a user ID.

The actual time of using a password is during program development, so this interface is merely a demonstration. You will find an application in the example of authentication.

Password: Test

Here you can test the keys selected under password parameters. Since passwords can be limited to certain ASCII ranges, e.g. capital letters, only the respective types are accepted.

In case the password entered is too simple a warning is displayed. Please do not expect a lot of brains in the deciding algorithm. It simply calculates statistical values like correlation coefficient and variance.

Passwords must be at least 4 characters long. Sometimes the result of encryption is not very persuasive because the same letters in different positions are coded into the same letters. (Actually, this is a flawless encryption since an unauthorized person does not know the original text had repeating letters!) The reason is not in the method used but in the fact that for demonstration the **ShareMan** uses only a fraction of the keys possible. The option "all characters" may yield non-printable results, but this does not affect evaluation in a program. In practice, you will hardly ever need this option.

You will find an application in the example of authentication.

Password: User ID

The user ID is a special form of the password. A customer's name is coded into a 5 digit number using the keys selected under password parameters. This number cannot be decoded back into the name, thus avoiding the typical problem of all crypto methods: If coded and non-coded text are displayed at the same time the code can easily be manipulated.

The name to be coded must be at least 5 characters long; the user ID always has exactly 5 digits. You may enter any character; the length of your entry is basically free but limited under the **ShareMan**'s interface.

You will find an application in the example of authentication.

Address file: List

This menu item displays the customer data stored. The **ShareMan** adds up the fees paid and shows the total sum.

With the left/right buttons you can leaf through the columns. The display jumps to the first record marked or to the beginning of the list if you have not marked anything. Since the Windows 3.1 listbox used here to contain the data only accepts 64 kByte you'll have to switch between record blocks via the up/down buttons if you have more than 880 entries.

With three buttons you can edit, delete and search for entries; editing can also be done with a double click. Marking entries to be deleted is done like in Windows' file manager, the number of entries marked is limited to 300 for memory reasons. A search always starts at the first entry and then from every match found down to the last entry, until the Search window is closed or you click the "From Top" button. If you switch the display to another 64 kByte block via the up/down buttons you will not see the match found; the search goes on correctly, however.

Selected entries can be printed directly. The number of entries marked is limited to 300 for memory reasons. Remember to set your printer to landscape format in the Windows control to fit all columns onto paper (The colums not displayed will be printed, too!). As an alternative, you can also export selected entries to the clipboard as ASCII text and from there import it into, let's say, a data base. You might have to reformat date and time columns. The semicolon serves as column limiter, the number of entries marked is limited to 100.

Hint: If you want to export more than 100 entries via the clipboard simply call this feature several times, importing every block in between. In case your computer cannot even cope with these 100 entries try to get more memory by closing other applications.

You can display your fees graphically to analyze when the most money came in and compare this to your marketing efforts.

See also Address file.

Address file: Analyze fees

Here you can display your fees graphically. They are shown cumulative, i.e. summed up over the time. Information is gathered from the customer data. Date range is from release date to the latest customer entry.

The **ShareMan** sets the release date to the day the parameter file was created. You can set it manually to every day from January 1, 1994, to December 31, 1999 to analyze the effect of your marketing efforts or the release of a new version. Instead of the market release date you can, of course, select any other.

The date format used does not depend on your system settings but on the customer data themselves, thus enabling both the German and the English version of the **ShareMan** to read and write both German (DD.MM.YY) and English (MM.DD.YY) dates correctly.

See also Address file.

LHA manager: Select archive

Here you select the archive, i.e. the file that is to contain the compressed files. The suggested file name extension is "LZH". If you want to create a self-extracting archive you still have to create an LZH arcive first.

The archive need not be in the same directory with the files to be compressed. It automatically becomes the source file, thus clearing the selection of more files. Note that the variables cannot be recognized anymore after compression.

See also LHA manager.

LHA manager: Add

Here you can select up to 50 files to be added to the archive. They can come from any directory; their path names, however, will not be stored (see LHA manager). Marking is done like in Windows' file manager. If you want to add files from different directories to the archive simply select this menu item several times.

The actions offered mean the following: *Add* copies the files to the archive, *Move* moves them, i.e. deletes the original files afterwards. *Update* works like *Add* but overwrites any older versions of the files that might already be in the archive. *Freshen* works like *Update* but only considers those files already present in the archive, i.e. you cannot expand the archive with newly created files. Your most frequent action will therefore be *Update*, so this is pre-selected.

Remember that LHA is a DOS program. The call from Windows might take a few seconds.

See also LHA manager.

LHA manager: Delete

This menu item deletes files from the archive. Marking is done like in Windows' file manager. If no files remain in the archive LHA automatically erases it.

Remember that LHA is a DOS program. The call from Windows might take a few seconds.

See also LHA manager.

LHA manager: Extract

This menu item get files back from the archive. Marking is done like in Windows' file manager. The files in the archive are not deleted.

Remember that LHA is a DOS program. The call from Windows might take a few seconds.

See also LHA manager.

LHA manager: Contents

This menu item displays the files contained in the archive, including information on compression.

Remember that LHA is a DOS program. The call from Windows might take a few seconds.

See also LHA manager.

LHA manager: Self-extracting

This menu item converts an LZH archive to a self-extracting "EXE" file, i.e. for decompression you will not need LHA anymore. The LZH archive remains intact; the EXE archive can be treated like an LZH archive.

Remember that LHA is a DOS program. The call from Windows might take a few seconds.

See also LHA manager.

Crypto: Cesar method

This simple method has been used by Cesar already who just moved all the alphabet's letters by three positions to the front. This implementation adds a square and a linear term to the constant offset. If this was too mathematical: The movement in the alphabet is not constant as it was with the old Roman but increases or decreases with every character, following a certain formula to be not too obvious.

For encryption and decryption of files three keys, each in the range from -16 to +16, can be selected, referring to constant, linear and square factor. The **ShareMan**'s interface uses only a fraction of the keys possible (see Using crypto algorithms in C programs).

You can encrypt and decrypt any type of file. The file name extension is changed to "ENC" for encrypted and "DEC" for decrypted files.

See also Crypto.

Crypto: Shift register method

This method generates a pseudo random sequence of numbers via a shift register. Most random number generators in digital computers work this way. The shift register initially contains an arbitrary sequence of bits (start value), two of which will be taken and put into a NOR gate. This gate's output is put back to the register's left end (loop) and the whole register is shifted one bit to the right, dropping the right-most bit. Regarding the whole register as a number yields the random value. This process is repeated with every character to be encrypted or decrypted.

For identical start values the sequence generated is also identical. The period length, i.e. the number of steps until the sequence repeats itself, is of great importance. The **ShareMan** uses an 8 bit shift register, usually resulting in period lengths of 15. This will do fine for the normal user. In case the text length is not greater than the period length *this crypto method is completely safe!*

For encryption and decryption of files three keys can be selected: the start value of the shift register in the range 00 to FF (hex) and the two loops in the range 0 to 7, which should not be identical. All three keys influence the period length. See also Using crypto algorithms in C programs; there you will also find a hint to a routine for calculating the period length.

You can encrypt and decrypt any type of file. The file name extension is changed to "ENC" for encrypted and "DEC" for decrypted files.

See also Crypto.

Crypto: Editor

This menu item starts Windows' editor "NOTEPAD.EXE" so that you can create files to be encrypted or view the results of encryption and decryption. The latest encryption or decryption result is loaded automatically. This does not include files that have been loaded to the editor manually.

See also Crypto.

Crypto: Encrypt

For encryption binary and text files (see Crypto) are distinguished, just like for decryption. The former are coded continuously (byte by byte), the latter line by line, i.e. leaving *Carriage Returns* intact. The keys are selected under Cesar method and shift register method; the method selected most recently is used.

Line by line encryption is useful when the text to be encrypted is actually a sequence of strings. The text in the **ShareMan**'s "About..." dialog is an example for that: The original strings were copied into a file that was encrypted line by line; the result was then copied back line by line into C source code strings to be decrypted by the program. For further information see Using crypto algorithms in C programs.

You can encrypt any file. The **ShareMan** suggests the file name extension "TXT" which you can override. The encrypted file gets the extension "ENC".

See also Crypto.

Crypto: Decrypt

For decryption binary and text files (see Crypto) are distinguished, just like for encryption. The former are coded continuously (byte by byte), the latter line by line, i.e. leaving *Carriage Returns* intact. The keys are selected under Cesar method and shift register method; the method selected most recently is used.

You can decrypt any file. The **ShareMan** suggests the file name extension "ENC" which you can override. The decrypted file gets the extension "DEC".

See also Crypto.

Example: Customer disk

This example will show you step-by-step how to get from a program to be distributed to a complete customer disk. All you need is a formatted disk the contents of which can be erased.

Step 1

Create a directory "C:\TEST" for test purposes and copy a number of arbitrary files into it. Using an editor, create a file "README.TXT" of arbitrary contents and another named "MYPROG.EXE" (regardless of file name extension it is to be a pure ASCII text) containing the variables for name, company and serial number anywhere. The latter must match the pattern "A00", i.e. be made up of a capital letter and two digits, with the authentication digit at the end. Save these two files to "C:\TEST".

In the real case "MYPROG.EXE" would be your program to be distributed and "README.TXT" an accompanying file. "C:\TEST" would be the path name both could be found under.

Step 2

Make "MYPROG.EXE" the source file and "A:\" the destination directory, also select "README.TXT" as more files. Under activities mark the first five offered.

Step 3

Now you have to adapt the serial number parameters to those mentioned above, i.e. length 3, pattern "A00" and control digit put to end. All characters are positive. The number to start with must be "A01". The **ShareMan** tells you that this is consecutive number "1".

In the real case this would mean you keep copy "A00" for yourself and expect 99 customers, but reserve the "A" as a counting position in case your expectations were too humble. As long as you have only few customers it enhances the serial number a little.

Step 4

The preparations which might have seemed cumbersome are done now. To avoid doing them again for this project save the parameter file in a directory of your choice by the name "MYPROG.SHM".

Step 5

Insert a disk with dispensable contents into drive "A" and select Go!. The **ShareMan** gets going and displays all activities in the status bar. In between you will be prompted for customer data. Name and company are written to disk; all data are put into the address file that is added to the parameter file.

In the real case this would be the disk to be delivered to your first customer.

Step 6

Exit the **ShareMan**, saving the changes, and check with an editor that serial number with authentication digit (a 3 in this case) as well as customer's name and company have been inserted into "MYPROG.EXE" in path "A:\". Of course, the control digit is too obvious for a serial number as simple as this.

Step 7

Start the **ShareMan** again and open your parameter file "MYPROG.SHM". Assure yourself in the overview that the settings are correct and under Serial Number: Start With... that it has been updated to "A02", meaning consecutive number "2".

Step 8

With the same disk in drive "A" select Go!. Specify other customer data.

In the real case this would be the disk for your second customer. Here we have overwritten the one for your first customer, of course.

Step 9

As in step 6, check the **ShareMan**'s work, and display your customer data. In the lower right your total fees are shown which you can also analyze graphically. (However, here you will meet the problem that all your fees are from the same day. If you still want to see the graphics set the second customer's date to the next day with an editor; see Structure of parameter files.)

Example: Authentication

A possible application for the user ID is inserting a check routine into your program in connection with an accompanying "INI" file or an entry in "WIN.INI". The **ShareMan** uses this method, starting from version 1.5.

Let's assume your customer's name is "Bernd Cordes". Bring this name into the form "BCORDES" to avoid problems with various lengths of first names. The **ShareMan** calculates a 5 digit user ID for you, let's say 83265. Calculation is done manually under the menu item User ID or automatically by selecting the respective activity.

Insert the lines "Name=BCORDES" and "ID=83265" into the "INI" file with the help of the appropriate Windows API commands. Your program reads both lines upon start and recalculates the ID from the name with the same password parameters (see Using crypto algorithms in C programs). If both numbers match the registration is okay. In case someone changes the name the ID will match no more. Display the name as "B. Cordes" somewhere in your program, e.g. the "About..." dialog or the title bar.

Note that with this method you can obtain individual copies for every customer without delivering a disk. The lines for name and ID are enough! Of course you can adapt serial numbers or any other data in the same manner.

If you still want to deliver a customer disk you can write the "INI" file to disk (destination directory) in Windows standard format by selecting the respective activity.

Using crypto algorithms in C programs

The include file "CRYPTO.H" declares 8 encryption and decryption functions to work on a single character, a string, a binary file, or a text file (for the difference between a binary file and a text file see Crypto.), furthermore a special method for calculation of a user ID with two auxiliary functions and a routine for the period length of the shift register method. The corresponding source code is "CRYPTO.C"; the foundation is ANSI-C.

For unexperienced C programmers: "CRYPTO.H" must be included in the C source code to use the algorithms via an *#include* command, "CRYPTO.C" must be added to your project or make file. If you find this too cumbersome simply copy both files into your source code. If you work in C++ make sure that "CRYPTO.C" is also compiled as C++. In Turbo and Borland C there is a compiler option *always C++*.

Hints on the expected parameters are in "CRYPTO.H".The modern form of parameter declaration in functions is used, i.e. inside the parentheses. Very old compilers might not swallow this.

Important: Encrypted text might contain the "\" character. In C, this is a special control character. If you want to copy encrypted strings directly into your source code change it to "\\" by hand. Use the *Search/Replace* option of your editor. The same is true for the string limiter character in case you want it to appear inside string limiters.

The three keys of the Cesar method can not only be in the range -16 to +16 as in the **ShareMan** but in the range -32768 to +32767. Thus you have 2^{48} possibilities, that is 281 trillions. Successfully tested, however, are only the keys in the range -256 to +256, that is 2^{27} possibilities or 134 millions. Exception: The user ID allows only two keys, that is 4 billions or 262 thousands possibilities, respectively. The shift register method offers 256 x (8 + 7 ... + 1) / 2, that is 4608 possibilities, because the two loops are commutative (meaning it is the same whether you specify 4 and 7 or 7 and 4).

Hint: A so-called bijective method can regain the original text from the encrypted text, knowledge of the correct keys provided, and vice versa. All crypto functions in "CRYPTO.H" and "CRYPTO.C" except *IDCalculate* are bijective; the latter is not so obvious in return.

Legal information: There are no restrictions for the use of the C routines in "CRYPTO.H" and "CRYPTO.C". This is also true for the unregistered version.

See also the examples of customer disk and authentication.

Structure of parameter files

Here you see an extract from the provided parameter file "EXAMPLE.SHM" with corresponding variable names:

```
0                   Copy
0                   Serial
1                   Customer
0                   Name
0                   Firm
0                   Calc
0                   Ini
c:\autoexec.bat     SourceFile
a:\                 DestinationDir
1                   MoreFiles->Items
config.sys          MoreFiles[i]
...
-1                  WriteFilePosSerial
-1                  WriteFilePosName
-1                  WriteFilePosFirm
4                   SerialLength
0                   SerialFrontOrEnd
1                   SerialType[i]
A                   SerialChar[i]
0                   SerialFactor[i]
...
22                  SerialNext
-2                  PassKey1
13                  PassKey2
3                   CesarKey1
-2                  CesarKey2
0                   CesarKey3
173                 ShiftKey1
1                   ShiftKey2
0                   ShiftKey3
0                   EnglishDates
01.01.94            ReleaseDate
21                  CustomerFile->Items
---CustomerData---
1                   CustomerFile->Number[i]
Cordes, Bernd       CustomerFile->Name[i]
Navy University     CustomerFile->Firm[i]
Wiesingerweg 34     CustomerFile->Street[i]
20253               CustomerFile->Zip[i]
Hamburg, Germany    CustomerFile->City[i]
0049 (40) 6547151   CustomerFile->Phone[i]
```

```
02/10/94            CustomerFile->Date[i]
18:06               CustomerFile->Time[i]
2A001               CustomerFile->Serial[i]
99999               CustomerFile->ID[i]
30                  CustomerFile->Money[i]
Author of ShareMan  CustomerFile->Remark[i]
...
```

Explanation:

Copy ... Ini	selected activities (0 = no, 1 = yes)
SourceFile	source file
DestinationDir	destination directory
MoreFiles...	count and names of additional files
WriteFilePos...	write position in destination file relative to beginning of file
SerialLength	length of serial number
SerialFrontOrEnd	position of control digit (0 = front, 1 = end)
SerialType	type of corresponding position (0 = digit, 1 = capital letter etc.)
SerialChar	fixed character in corresponding position
SerialFactor	value of corresponding position
SerialNext	next serial number as consecutive number
PassKey	two password keys
CesarKey	three Cesar keys
ShiftKey	three shift register keys
EnglishDates	indicator for English date format (0 = no, 1 = yes)
ReleaseDate	release date
CustomerFile...	address file
Number ... Remark	customer data

Regardless of the serial number's actual length all 12 positions possible are stored. Similarly, a fixed character is stored even if in the corresponding position there is none specified. The factor of positive positions is +7, the factor of negative positions is -3.

By the indicator for English date format both German and English parameter files can be read and written by both the German and the English version of the **ShareMan**. Settings for the date format in the Windows control are ignored.

Blank fields in the customer data are represented by blank lines.

Shareware

This is an unregistered version of the **ShareMan**. The only drawback in comparison to the registered version is a reminder screen upon program start. Thus usability is only slightly reduced.

This program and its accompanying documentation are Shareware. The Shareware principle states that you may use the product for a limited time or to a limited extent freely, on violation of these limits, however, either erase the product completely from your system or convert it into the registered version by paying the registration fee, then being subject to normal copyright rules without the limitations mentioned. In the case of the **ShareMan** the limitation is that you must not use the unregistered version for more than 30 days. Remember: Trust is the foundation of the Shareware principle!

All users of the unregistered version of the **ShareMan** are granted a limited license to copy the product only for the trial use by themselves or others, provided that the **ShareMan** is copied in its full and unmodified form. The copy must include all files necessary to permit full operation of the program. It must also include this documentation. Use of unregistered copies of the **ShareMan** by any person in connection with a business, corporation, educational institution, or government agency is forbidden. Such users must register this product. By using the unregistered version of the **ShareMan** you acknowledge that you have read and understood this limited license and agree to be bound by its terms and conditions. The registered version must not be distributed.

Registration / Contact address

This is an unregistered version of the **ShareMan**. If you use the **ShareMan** for more than 30 days you will have to register. To do so, switch on your printer and then click on this text: Order form.

Bernd Cordes
Wiesingerweg 34
D-20253 Hamburg
Germany
Phone/Fax +49 (40) 494370
CompuServe: 100334,375
Internet: 100334.375@compuserve.com

As a registered user you will receive at no cost a user ID to convert your unregistered version into a registered version. Because of the context sensitive online help I do without a printed manual. Your personal version may be used without restrictions other than those resulting from copyright laws. Support by phone, fax, mail and e-mail is included.

Please see also the legal information.

Please print this **Order form** (`File / Print Topic`), fill it in and mail/fax it.

Bernd Cordes
Wiesingerweg 34
D-20253 Hamburg
Germany
Fax +49 (40) 494370

Name: _____ Company: _____

City: _____ Zip code: _____

Street: _____ Country: _____

Phone: _____ Fax: _____

I order:

_____ licenses ShareMan 1.7 english

credit card, check: DM 40 each DM _____ total

all others: US$ 25 each US$ _____ total

Site licenses available; please inquire.
Prices include taxes and shipping & handling. Credit card or check users, please check your current exchange rate.

I pay:

O cash; money is enclosed

O MasterCard/Access no _____ exp. date _____

O Visa card no _____ exp. date _____

O CompuServe Registration Base (GO SWREG) No. 2936

O check (drawn on a German bank in DM, payable to Bernd Cordes)

O international postal money order

_____
Signature

If you can spare another piece of paper, please take the time to answer the following questions. You help to rid this product of possible bugs and to serve the customers' needs better.

Supplier of ShareMan 1.7:

- O Shareware distributor _____
- O CD-ROM _____ published by _____
- O CompuServe, forum _____
- O MSN, provider _____
- O other information service: _____
- O Internet, FTP server / WWW page _____
- O BBS _____ in _____
- O something totally different: _____

Opinion in school marks:

usefulness:	A	B	C	D	E
user friendliness:	A	B	C	D	E
readme file, online help:	A	B	C	D	E
maturity / crash safety / bugs:	A	B	C	D	E

An F is not provided - if my program were that bad you would hardly have made it through here.

Suggestions for improvements:

Thank you for your effort!

Legal information

Beside defining the user rights for your unregistered version of the **ShareMan** the following information is necessary:

The author of the **ShareMan** is Bernd Cordes, Wiesingerweg 34, 20253 Hamburg, Germany. He owns the unlimited copyright. You are granted permission to use program and documentation on one computer at the same time. The unregistered version may be distributed freely.

Limited Warranty:
This product is provided as is, without any representation or warranty of any kind, either express or implied, including without limitation any representations or endorsements regarding the use of, the results of, or performance of the product, its appropriateness, accuracy, reliability, or correctness. The entire risk as to the use of this product is assumed by the user. The author does not assume liability for the use of this program beyond the original purchase price of the software. In no event will the author be liable for additional direct or indirect damages including any lost profits, lost savings, or other incidental or consequential damages arising from any defects, or the use of or inability to use this program, even if the author has been advised of the possibility of such damages. Some US states do not allow the exclusion or the limit of liability for consequential or incidental damages, so the above limitation may not apply to you.

Distribution Policy:
Permission is granted to any individual or business to distribute the unregistered copy of the **ShareMan**, provided they convey a complete and unaltered copy of the entire package (recompression or decompression is permitted). This permission pertains to any kind of data media including online services and BBSes, any form of packaging and bundling, and any distribution fee. The author may revoke any permissions granted here by notification in writing.

MS-DOS and Windows are trademarks of Microsoft Corporation.
Borland C++ is a trademark of Borland International.

ASP ombudsman statement

This program is produced by a member of the Association of Shareware Professionals (ASP). ASP wants to make sure that the shareware principle works for you. If you are unable to resolve a shareware-related problem with an ASP member by contacting the member directly, ASP may be able to help. The ASP Ombudsman can help you resolve a dispute or problem with an ASP member, but does not provide technical support for members' products. Please write to the ASP Ombudsman at 545 Grover Road, Muskegon, MI USA 49442-9427, Fax 616-788-2765, or send a CompuServe message via CompuServe Mail to ASP Ombudsman 70007,3536.

Other products

Depleter 2.2:
Windows utility for owners of laptops and notebooks.
Countdown mode for controlling remaining battery use time, based on previous measurements during your normal work. Thus you can see long before your computer's built-in voltage control when you will have to shut down your work.
Discharge mode for calculated battery depletion under constant conditions. Thus you can detect and cure the memory effect, i.e. the steady decline of maximum battery capacity.
Available in English and German. US$ 10 or DM 20.

Backer 2.8:
Windows utility for comparing, synchronizing and updating files between any kind of data media, also via network, disk, or cable. Uses file creation dates to compare. Interactive selection of included or excluded files, file types and directories. Several options for sorting, path reduction, confirmation and filtering. Takes parameters for automatic work. Almost unlimited file quantities.
Available in English and German. US$ 25 or DM 40.

Good Credit 1.2
Windows utility for verifying credit card numbers. Supports Visa, MasterCard/Eurocard, American Express, Discover, and others.
Available in English and German. US$ 10 or DM 20.

About the author

I was born in 1966 in little Stade near Hamburg. My first steps in the world of computers were on a TI 99/4A. Via Atari 400, Atari 130 XE, Atari 800 XL and Atari 1040 ST I climbed up to the PC in 1992, which turned from a mere writing tool to a fascinating programming machine, thanks to Borland C++ 3.1.
In 1986, I joined the Federal German Navy, doing an 11-month trip around the world on a sailing ship. Deciding the navy was the right place for me I applied for career officer, including a university degree in communications engineering, with a focus on digital electronics, microprocessors, and C programming.
In private, I am a passionate bachelor and now and then enjoy the multi-faceted nightlife of Hamburg.

_____
Bernd Cordes, 11/24/1995