binhex

nsoggia@telnetwork.it

COLLABORATORS			
	TITLE : binhex		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	nsoggia@telnetwork.it	May 24, 2025	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

binhex

1.1 The Nik Soggia BinHex Toolkit

The Nik Soggia _____ _____
_____ _____
_____ _____
_____ _____
_____ _____

"Easy to use BinHex 4.0 encoding and decoding programs for the Amiga"

BHD v37.02 (18-Oct-96) - BHE v37.02 (18-Oct-96) _____

Foreword About programs and BinHex format

BHD Decoding BinHex files using BHD BHE Encoding BinHex files using BHE

Tech notes Technical matters for I/O files History Programs' history

DISTRIBUTION

This package is released under the concept of freeware, the package must be distributed as one whole. The distributor may charge a fee up to the cost of the medium for the entire package.

NO WARRANTY

This package is provided as is, without warranty of any kind, either expressed or implied. Should the package prove defective, you assume the entire cost of all necessary servicing, repair or correction even if I have been advised of the possibility of such damages. I'm not responsible of the results of the use of the package.

SUPPORT

If you have any suggestions, bug reports, or wish to let me know something about the package feel free to contact me at nsoggia@telnetwork.it

1.2 Foreword on the BinHex Toolkit

ABOUT MAC FILING SYSTEM

Mac files consist of two parts called forks: the data fork is the most important for non-Mac users as it contains the meaningful data, the resource fork contains data such as program segments information, the icon bitmap and other arbitrary data that can resemble Amiga's icon tooltypes. Then, each Mac file has some more additional information stored in a hidden file called the "desktop database". Transferring such complicated files from one Mac to another without using Mac specific media needed a way to keep together all the above information, so BinHex was born. This format is now so popular that Mac users use it to send

their data even to other platform users who basically need only the data fork. Even Windows users seem to like it as their default encoding format: "it makes bigger files so it must be good" =8-O

ABOUT BINHEX 4.0

BinHex 4.0 merges the original file name, the desktop database information, the data fork and the resource fork in a single file, then compresses repeated bytes and encodes the binary data in printable ASCII to safely transfer it by e.g. E-Mail. It also provides an error checking mechanism to check if the file got corrupted during the transfer.

BINHEX VS BASE64

BinHex is essential only when data has to be exchanged between Macs, In all other cases Base64 encoded MIME messages can do the same things BinHex does with less CPU overhead, less bandwidth and improved inter-platform compatibility.

AMIGA IMPLEMENTATION

The BinHex Toolkit is able to preserve all the information a BinHex 4.0 file can contain:

FILE NAME: Amiga file names can be up to 30 characters in length, Mac names can be up to 63 characters long. When importing, the BinHex Toolkit translates the Mac character set to 7-bit ASCII, strips unprintable characters, eventually truncates the name length to 30 characters. When exporting, the BinHex Toolkit translates the Amiga character set to 7-bit ASCII.

FILE TYPE: The BinHex Toolkit stores (and searches) the Mac file type in the Amiga file comment.

FILE CREATOR: The BinHex Toolkit stores (and searches) the Mac file creator in the Amiga file comment.

FINDER FLAGS: The BinHex Toolkit stores (and searches) the Mac Finder flags (bozo, system, bundle, invisible, locked) in the Amiga file comment.

DATA AND RESOURCE FORKS: The BinHex Toolkit reads and writes an Amiga file for each fork in the BinHex file.

AMIGA VS MAC

The BinHex Toolkit is faster, more fault tolerant, compresses better, needs less resources, has more options than the original Mac 4.0 binary (no kidding, I can demonstrate it). It lacks the GUI, but can be interfaced easily with mail and news readers.

1.3 BinHex_decode (BHD) User's manual

BHD stands for BinHex Decoder, it accepts a BinHex 4.0 encoded file and generates up to two files containing the data and resource forks.

REQUIREMENTS

BHD runs under Kickstart 2.0 and above from any Shell (it does not support Workbench directly) on any Amiga, it needs at least 4 Kb of stack memory and at least about 25 Kb of free memory.

USAGE

INFILE/A This required argument is the name of the input file (wildcards are not supported). The file must be in BinHex format.

DATA/K This keyword argument designates the file name where the data fork will be stored. If the argument contains a "%s" token (without quotes), the token will be replaced with the filename stored in the BinHex file plus a "_d" suffix. Only one "%s" token can be specified, specifying more than one token will lead to randomly messed up names.

RES=RSRC/K This keyword argument designates the file name where the resource fork will be stored. If the argument contains a "%s" token (without quotes), the token will be replaced with the filename stored in the BinHex file plus a "_r" suffix. Only one "%s" token can be specified, specifying more than one token will lead to randomly messed up names.

BUFSIZE/K/N This keyword numeric argument designates the size in bytes of each work buffer. The buffers are 3 and equal in size, allocated in a contiguous chunk of memory. Minimum value is 8192, maximum value depends on the largest free memory chunk in your system, default size for each buffer is 16 Kb.

NOTE/S If this switch is set then each output file will be commented with the file type, the file creator and the Finder flags. See also the file comment format chapter.

OPERATION

INFILE doesn't need to start with a BinHex header string, BHD will scan the whole file until it finds one. If INFILE contains more BinHex documents, only the first one will be extracted.

The data and resource forks will be extracted only if their name has been specified by the DATA/K,RES=RSRC/K arguments: in this way you can extract only the forks you really need (most of the times the data one). BHD will never create empty files: if you need empty files, you can generate a couple of empty files and let BHD overwrite them.

Using a large (100 Kb or more) BUFSIZE can improve decoding time when reading or writing large files from CD-Rom's or similar relatively slow media, but it needs much memory and uses intensively the CPU for larger time intervals.

The NOTE switch allows to write the output files directly on Mac media (such as ShapeShifter's mac-handler) or simply to preserve the Mac file attributes for a future re-encoding: it comments the output files with the type/creator pair followed by the active Finder flags. Should the type/creator pair be invalid, the ????/???? default will be used.

ERROR MESSAGES

BHD shows the standard AmigaDOS messages on its standard output in case of file I/O error or memory allocation failure. BHD will set return code to FAIL (20) if something goes wrong, ERROR (10) if command line options are wrong, OK (0) if all goes flawlessly.

[INFILE]: not a hqx7 file Infile hasn't the BinHex header string "(This file must be converted with BinHex 4.0)" or its data doesn't start with a ":", or the header string is not separated by at least a "end of line" character (ASCII CR or LF) from the encoded data, or the file header and data are indented from the left margin. You can recover the file removing the indentation in front of the header line and the first data line, or adding the missing header string, ":" or end-of-line characters.

[INFILE]: unexpected end of file Infile hasn't a ":" character at the end of its data chunk. If you add one by hand after the last data character and then you get a CRC error, then the file really got truncated and there is no way to recover lost data.

crc error in [HEAD|DATA|RSRC] (expected [XXXX], found [YYYY]) If the CRC error is in the BinHex header no extraction will be attempted. If the CRC error is in the forks, corrupted forks will not be deleted.

1.4 BinHex_encode (BHE) User's manual

BHE stands for BinHex Encoder, it generates a BinHex 4.0 encoded file reading up to two files containing the data and resource forks.

REQUIREMENTS

BHE runs under Kickstart 2.0 and above from any Shell (it does not support Workbench directly) on any Amiga, it needs at least 4 Kb of stack memory and at least about 25 Kb of free memory.

USAGE

OUTFILE/A This required argument is the name of the BinHex output file.

DATA/K This keyword argument designates the file name containing the data fork.

See also the file comment format chapter.

RES=RSRC/K This keyword argument designates the file name containing the resource fork.

See also the file comment format chapter.

BUFSIZE/K/N This keyword numeric argument designates the size in bytes of each work buffer. The buffers are 3 and equal in size, allocated in a contiguous chunk of memory. Minimum value is 8192, maximum value depends on the largest free memory chunk in your system, default size for each buffer is 16 Kb.

NOTE/S If this switch is set then the output file will be commented "TEXT/BNHQ". See also the file comment format chapter.

MIME/S If this switch is set then the appropriate MIME header will be prepended to the BinHex header string.

CR/S If this switch is set then each line will be terminated by a Carriage Return character (Mac end of line marker) instead of a Line Feed character (Amiga end of line marker).

APPEND/S If this switch is set then the data generated by the program will be stored starting at the end of the already existing outfile instead of overwriting the outfile.

OPERATION

The OUTFILE/A value will be used also as the default BinHex file name. It will be stored in the BinHex header in 7 bit ASCII format, if the value has a 3 characters extension to the right, the extension will be stripped (examples of such extensions may be: ".out", ".hqx", ".mac", or any other string starting with a "." followed by 3 characters) so a outfile named "Test.hqx" will save the data in a file called "Test.hqx" and will generate the "Test" default BinHex file name, and a outfile named "Test.hqx7" will save the data in a file called "Test.hqx7" and will generate the "Test.hqx7" default BinHex file name.

The data and resource forks will be saved only if their name has been specified by the DATA/K,RES=RSRC/K arguments: in this way you can generate BinHex files containing just one fork, both forks, or no forks at all.

The NOTE and CR switches are designed to write directly the output file directly on Mac media (such as ShapeShifter's mac-handler).

The MIME and APPEND switches can be used when invoking BHE from scripts to compose E-Mail bodies, note that the MIME name preserves the 3 characters file name extension if present.

Using a large (100 Kb or more) BUFSIZE can improve encoding time when reading or writing large files from CD-Rom's or similar relatively slow media, but it needs much memory and uses intensively the CPU for larger time intervals.

When getting the desktop database information from the file comment, Finder flags will be OR-combined, so if a flag is present in the data fork file comment and another flag is present in the resource fork comment, the resulting BinHex header will contain both flags. When getting file type and file creator from file comments, BHE tries to use the ones stored in the data file comment, if the data file is not specified, not found, or its comment is not set, BHE will check the resource fork file, if the resource file is not specified, not found, or its comment is not set a default ????/???? type/creator pair will be set in the BinHex header.

ERROR MESSAGES

BHE shows the standard AmigaDOS messages on its standard output in case of file I/O error or memory allocation failure. BHE will set return code to FAIL (20) if something goes wrong, ERROR (10) if command line options are wrong, OK (0) if all goes flawlessly.

1.5 Technical matters

FILE COMMENT FORMAT

File comment contains strings separated by white spaces. The first string that is 9 characters long and has a "/" character in the middle will be considered the file type/creator pair, if the 5th character of the comment string is a "/" then no further check will be made and the first 9 characters of the comment will be considered the type/creator pair (so that type and creator may contain spaces). All the 3 characters long strings will be (case insensitively) matched with "loc", "inv", "bun", "sys", "boz" to set active the "locked", "invisible", "bundle", "system" and "bozo" Finder flags.

WHAT FINDER FLAGS NEED TO BE SET

Leave them unset, as they may sound familiar they are not. The only purpose BHD stores those flags in the file comment is to allow BHE to rebuild the same BinHex file BHD decoded...

1.6 History of the BinHex Toolkit

DEVELOPMENT

I write and test the BinHex Toolkit on an A3000/030-25, Before releasing each new version to the public I test the program under Enforcer and Mungwall on Kickstart 3.1.

RELEASE HISTORY

BHD 37.01 (10-Oct-96) First public release. BUG: BHD can't accept more than one "%s" token for each output file name format string but it does not check for multiple tokens.

BHE 37.01 (10-Oct-96) First public release. BUG: file comment interpreter doesn't support spaces in type/creator strings.

BHD 37.02 (18-Oct-96) Second public release. BUGFIX: v37.01 failed with "not an hqx7 file" error if the BinHex header was not found within the first BUFSIZE/K/N bytes of the INFILE/A. BUGFIX: v37.01 had a flacky "%s" token interpreter. I rewrote the token handling routine, now it ignores extra "%s" tokens. ADDED: "%s" token is replaced with default BinHex name plus "_d" or "_r". "%S" token is replaced with plain default BinHex name.

BHE 37.02 (18-Oct-96) Second public release. CHANGED: made a smarter file comment parser, type/creator can now contain spaces if the type/creator string starts at the beginning of the file comment.