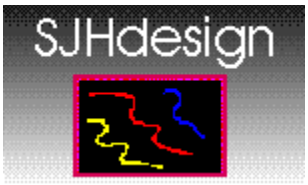

http://www.sjhdesign.com/

SJHb64 1.6

Help Index

Base64 Help

Menu Commands

File Menu

DECODE FILE - pops up a file open dialog box to open a base64 encoded file with any extension. The most common ones are .b64, .txt, .enc, .asc and .dat.

ENCODE FILE - also pops up a file open dialog which you can use to load any type of file for base64 encoding.

Options Menu

DIRECTORIES - This menu item has two subitems where you can choose TWO sets of default directories, one set for encoding and one for decoding.

DECODING - here you can specify the decoding input directory(from which you load base64 encoded files for decoding) and the decoding output directory, where your decoded files will be written to. If you are using SJHB64 as a decoding extension to a newsreader or web browser, it's important that you at least specify a decoding output directory. It's also helpful for those who use the Auto-Decode feature(see below) to specify default directories. Even if you don't use SJHB64 in this manner, most people like to use one directory for storing encoded files that we've downloaded and another directory for the decoded output of those files. Specifying these directories in SJHB64 always makes it easy to locate your files. In other words, when you specify a default input directory, every time you bring up the File Open dialog to load files, that's the directory the dialog will show. When you specify a default output directory, the Save As dialog will always show that directory and if you use the auto decoding option, your decoded files will automatically be written to that directory.

ENCODING - here you can specify the encoding input directory, where you load files for base64 encoding, and the encoding output directory, where your base64 encoded files will be written to.

NOTE : If no directories are specified by the user the program uses the most recently used directory as it's default input directory. For example, if you last loaded a base64 file from directory c:for decoding, the next time you choose DECODE FILE, the file open dialog will come up with c:as it's directory. If no output directory is chosen, the programs default behavior is to write decoded files to the same directory as the source file. The same goes for encoding - SJHB64's default behavior is to write an encoded file to the same directory as it's source file.

AUTO CLOSE - this feature was added those for who use the Windows ASSOCIATE command to launch SJHB64 and decode a file by clicking on a base64 file with a default extension(like .b64). The program will automatically shut down after the decoding operation is complete. This feature also comes in handy for those using SJHB64 as a decoding extension for a Newsreader or Web Browser-you don't have to manually close the program after each file is decoded.

AUTO DECODE - this choice allows the user to choose a file for decoding (via the DECODE FILE menu command or by dragging and dropping the file from Explorer or File Manager) and SJHB64 does the rest without user intervention. The file is automatically decoded to

the decoding output directory, if one is specified in advance with the DIRECTORIES menu choice, or else to the same directory as the source encoded file.

Loading Files for Encoding and Decoding

There are three ways to load files for encoding and decoding in SJHB64:

Drag and Drop

This is the fastest way to do it. Just drag a bunch of files from the Win 3X File Manager or Win95 Explorer, drop them on SJHB64 and decode or encode them all in one shot. If the Auto decode feature described above is enabled, you don't have to do anything-they're all decoded to your default directory. You can even combine files for encoding and decoding in the same batch and SJHB64 is smart enough to figure out which ones to encode and which ones to decode! If auto decode is not enabled, a dialog will pop up asking you to confirm the file name and location for your decoded files.

The File Open Dialog

For those who prefer the old fashioned way, choose either ENCODE FILE or DECODE FILE on the FILE Menu and a standard File Open dialog will pop up, allowing you to choose files for encoding or decoding.

The Windows ASSOCIATE Command

File Manager and Explorer both support the ASSOCIATE command, which allows you to launch SJHB64 and decode a base64 encoded file simply by clicking on the file. For example, if you would like to automatically decode .B64 files by double clicking them in File Manager or Explorer, highlight a file with the .B64 extension and choose ASSOCIATE from File Manager's FILE menu. A dialog will pop up, into which you will type the full path to SJHB64. Choose OK, and everytime you click on a file with the .B64 extension in File Manager, SJHB64 will be launched and your file decoded. Combine this with SJHb64's AUTO CLOSE and AUTO DECODE options and you have one click decoding-just double click on a .B64 file and it's automatically decoded and SJHB64 closes, all in one step!

BASE64 AND FILE EXTENSIONS:

There is no standard extension for base64\Mime files, which is a source of much confusion among users. While most uuencoded files have the .uue extension, base64 encoded data comes in various mime compliant and non mime compliant file types with many extensions like .txt, .asc, .b64 or no extension at all depending on the program used to encode and post the data. SJHB64 will accept a file with any extension and check to see if there is any Base64 encoded data in it. If there's none, the program will pop up a 'No data to decode' message and terminate the operation. Likewise, you can write base64 files with any extension you like-just type it into the filename field of the SAVE AS dialog.

NON MIME COMPLIANT BASE64 FILES:

There are many files on USENET that don't comply with the MIME standard for email exchange. The standard specifies that the file header should contain information about the file type, filename and extension, but you often run across USENET posts that don't have this information. In the past, these files have been a major source of frustration for users. SJHb64 tries to simplify this difficult operation for you - here's how:

Instead of just decoding files like this and creating a generically named file with no extension or popping up a dialog asking you to supply the file type(if you remember), SJHb64 analyzes the Base64 data to determine if the encoded file contains a 'recognized' data type and if it does, adds the extension for you. Since we also have to have a name to write the file, the name of the source base64 file is used by default. If the Auto Decode option is not used, the program will pop up a Save As dialog allowing you to change the name if desired-do NOT change the extension, though. For example, let's say that you have a base64 file named MYSTERY.B64 that contains a gif file but no mime header containing this information. After SJHb64 decodes it, the resulting gif will be in the same directory with the name MYSTERY.GIF. While that was probably not the original name of the gif file, it's impossible for SJHb64 to determine the real name if no Mime header data is present. These are the data types that SJHb64 can recognize in encoded form:

.JPG(JPEG image files)
.TIF(TIFF image files)
.BMP(Windows bitmap files)
.ZIP(PKzip compatible archive files)
.AVI(MS video files)
.MOV(Quicktime video files)
.PCX(PCX image files)
.EXE(executable files)
.WRI(Windows Write files)
.XLS(MS Excel files)
.GIF(GIF image files)
.EPS(Postscript printer files)
.MPG(Mpeg video files)
.WAV(MS sound files)
.MID(MIDI music files)
.VOC(VOC sound files)
.DLL(Windows dynamic link libraries)
.DOC(MS Word documents)
.PPT(MS PowerPoint presentation files)

This list does not include various subformats that are also recognized.

WINDOWS 95 LONG FILE NAMES-

At this writing, Windows95 long filenames are not fully supported, so if the MIME file you are decoding contains an encoded binary with a long filename, SJHb64 will truncate the name of the encoded file to something conforming to the DOS 8.3 filename standard. In general, the program will use the first 8 characters of the long filename to make a DOS compliant name. If any of those first eight characters contains a space, the program will insert a dash<-> where the space was. So

trytoloadthis.gif becomes trytoloa.gif

some long name.jpg becomes some-lon.jpg

If you need to know what the original long filename of the embedded binary was, you'll have to load the file into a text editor and look through the MIME header for a line containing:

Content-Disposition: inline; filename="some long filename.jpg"

or:

Content-Disposition: attachment; filename="some long filename.jpg"

If the filename contains any other non-dos-supported characters besides spaces in the first eight characters, SJHb64 will pop up a dialog announcing that it cancelled the save operation.

This applies even if you are running Windows 95. Of course, you can also use Explorer to restore the original long filename if you are running Windows 95.

MIME FILES WITH MULTIPLE BASE64 ENCODED SECTIONS:

The MIME specification supports multiple messages and binaries within the same file. These multiple encoded segments can also be of different types. A common example is what happens when you email your multimedia Word document containing linked BMP images and WAV sounds. Your mail program will often create one MIME file with a base64 encoded section for the Word document itself (content\type : application\msword), seperate encoded sections for each linked .BMP file (content\type: application\x-bitmap) and for each WAV file (content\type: audio\x-wav). If the recipient has a decoder like SJHb64 that supports multiple encoded segments within one MIME file, it's simple to decode- the program will pop up a Save As dialog as it encounters each file, and then write all the decoded files to the same directory so the recipient can easily load the document with all of it's linked files.

OTHER SOURCES OF MIME AND BASE64 INFORMATION:

MIME(the Multi-purpose Internet Mail Extensions) has made it possible for us to send E-Mail messages containing images, sound, video and many other binary formats, which are usually Base64 encoded. This MIME\Base64 relationship can be difficult for the average user to understand-MIME is not a file type, but a method of structuring text and encoded binary information in E-Mail messages so that the software on the other end of the line can understand and use it. Base64 is the method used to encode binary information(like images, sound, video, etc.) into ASCII text form so that it can pass through e-mail gateways, which cannot transport binary data. All this can be confusing, so it might help if you take a look at these additional sources of information:

The MIME FAQ - a regularly updated list of frequently asked questions about MIME, available on the comp.mail.mime newsgroup. We also keep a current copy of this FAQ at our site for downloading.

The Base64 Guide for USENET Types - our own guide to base64 decoding, available at our site:
http://www.sjhdesign.com/