USAGE instructions for the Independent JPEG Group's JPEG software

=================================================================

This file describes usage of the JPEG conversion programs cjpeg and djpeg,

as well as the utility programs jpegtran, rdjpgcom and wrjpgcom. (See

the other documentation files if you wish to use the JPEG library within

your own programs.)

If you are on a Unix machine you may prefer to read the Unix-style manual

pages in files cjpeg.1, djpeg.1, jpegtran.1, rdjpgcom.1, wrjpgcom.1.

INTRODUCTION

These programs implement JPEG image encoding, decoding, and transcoding.

JPEG (pronounced "jay-peg") is a standardized compression method for

full-color and grayscale images.

GENERAL USAGE

We provide two programs, cjpeg to compress an image file into JPEG format,

and djpeg to decompress a JPEG file back into a conventional image format.

On Unix-like systems, you say:

cjpeg [switches] [imagefile] >jpegfile

or

djpeg [switches] [jpegfile] >imagefile

The programs read the specified input file, or standard input if none is

named. They always write to standard output (with trace/error messages to

standard error). These conventions are handy for piping images between

programs.

On most non-Unix systems, you say:

cjpeg [switches] imagefile jpegfile

or

djpeg [switches] jpegfile imagefile

i.e., both the input and output files are named on the command line. This

style is a little more foolproof, and it loses no functionality if you don't

have pipes. (You can get this style on Unix too, if you prefer, by defining

TWO\_FILE\_COMMANDLINE when you compile the programs; see install.txt.)

You can also say:

cjpeg [switches] -outfile jpegfile imagefile

or

djpeg [switches] -outfile imagefile jpegfile

This syntax works on all systems, so it is useful for scripts.

The currently supported image file formats are: PPM (PBMPLUS color format),

PGM (PBMPLUS grayscale format), BMP, Targa, and RLE (Utah Raster Toolkit

format). (RLE is supported only if the URT library is available, which it

isn't on most non-Unix systems.) cjpeg recognizes the input image format

automatically, with the exception of some Targa-format files. You have to

tell djpeg which format to generate.

JPEG files are in the standard JFIF file format. There are other,

less widely used JPEG-based file formats, but we don't support them.

All switch names may be abbreviated; for example, -grayscale may be written

-gray or -gr. Most of the "basic" switches can be abbreviated to as little as

one letter. Upper and lower case are equivalent (-BMP is the same as -bmp).

British spellings are also accepted (e.g., -greyscale), though for brevity

these are not mentioned below.

CJPEG DETAILS

The basic command line switches for cjpeg are:

-quality N[,...] Scale quantization tables to adjust image quality.

Quality is 0 (worst) to 100 (best); default is 75.

(See below for more info.)

-grayscale Create monochrome JPEG file from color input.

Be sure to use this switch when compressing a grayscale

BMP file, because cjpeg isn't bright enough to notice

whether a BMP file uses only shades of gray. By

saying -grayscale, you'll get a smaller JPEG file that

takes less time to process.

-rgb Create RGB JPEG file.

Using this switch suppresses the conversion from RGB

colorspace input to the default YCbCr JPEG colorspace.

You can use this switch in combination with the

-block N switch (see below) for lossless JPEG coding.

See also the -rgb1 switch below.

-optimize Perform optimization of entropy encoding parameters.

Without this, default encoding parameters are used.

-optimize usually makes the JPEG file a little smaller,

but cjpeg runs somewhat slower and needs much more

memory. Image quality and speed of decompression are

unaffected by -optimize.

-progressive Create progressive JPEG file (see below).

-scale M/N Scale the output image by a factor M/N. Currently

supported scale factors are M/N with all N from 1 to

16, where M is the destination DCT size, which is 8 by

default (see -block N switch below).

-targa Input file is Targa format. Targa files that contain

an "identification" field will not be automatically

recognized by cjpeg; for such files you must specify

-targa to make cjpeg treat the input as Targa format.

For most Targa files, you won't need this switch.

The -quality switch lets you trade off compressed file size against quality of

the reconstructed image: the higher the quality setting, the larger the JPEG

file, and the closer the output image will be to the original input. Normally

you want to use the lowest quality setting (smallest file) that decompresses

into something visually indistinguishable from the original image. For this

purpose the quality setting should be between 50 and 95; the default of 75 is

often about right. If you see defects at -quality 75, then go up 5 or 10

counts at a time until you are happy with the output image. (The optimal

setting will vary from one image to another.)

-quality 100 will generate a quantization table of all 1's, minimizing loss

in the quantization step (but there is still information loss in subsampling,

as well as roundoff error). This setting is mainly of interest for

experimental purposes. Quality values above about 95 are NOT recommended for

normal use; the compressed file size goes up dramatically for hardly any gain

in output image quality.

In the other direction, quality values below 50 will produce very small files

of low image quality. Settings around 5 to 10 might be useful in preparing an

index of a large image library, for example. Try -quality 2 (or so) for some

amusing Cubist effects. (Note: quality values below about 25 generate 2-byte

quantization tables, which are considered optional in the JPEG standard.

cjpeg emits a warning message when you give such a quality value, because some

other JPEG programs may be unable to decode the resulting file. Use -baseline

if you need to ensure compatibility at low quality values.)

The -quality option has been extended in IJG version 7 for support of separate

quality settings for luminance and chrominance (or in general, for every

provided quantization table slot). This feature is useful for high-quality

applications which cannot accept the damage of color data by coarse

subsampling settings. You can now easily reduce the color data amount more

smoothly with finer control without separate subsampling. The resulting file

is fully compliant with standard JPEG decoders.

Note that the -quality ratings refer to the quantization table slots, and that

the last value is replicated if there are more q-table slots than parameters.

The default q-table slots are 0 for luminance and 1 for chrominance with

default tables as given in the JPEG standard. This is compatible with the old

behaviour in case that only one parameter is given, which is then used for

both luminance and chrominance (slots 0 and 1). More or custom quantization

tables can be set with -qtables and assigned to components with -qslots

parameter (see the "wizard" switches below).

CAUTION: You must explicitly add -sample 1x1 for efficient separate color

quality selection, since the default value used by library is 2x2!

The -progressive switch creates a "progressive JPEG" file. In this type of

JPEG file, the data is stored in multiple scans of increasing quality. If the

file is being transmitted over a slow communications link, the decoder can use

the first scan to display a low-quality image very quickly, and can then

improve the display with each subsequent scan. The final image is exactly

equivalent to a standard JPEG file of the same quality setting, and the total

file size is about the same --- often a little smaller.

Switches for advanced users:

-arithmetic Use arithmetic coding.

CAUTION: arithmetic coded JPEG is not yet widely

implemented, so many decoders will be unable to

view an arithmetic coded JPEG file at all.

-block N Set DCT block size. All N from 1 to 16 are possible.

Default is 8 (baseline format).

Larger values produce higher compression,

smaller values produce higher quality

(exact DCT stage possible with 1 or 2; with the

default quality of 75 and default Luminance qtable

the DCT+Quantization stage is lossless for N=1).

CAUTION: An implementation of the JPEG SmartScale

extension is required for this feature. SmartScale

enabled JPEG is not yet widely implemented, so many

decoders will be unable to view a SmartScale extended

JPEG file at all.

-rgb1 Create RGB JPEG file with reversible color transform.

Works like the -rgb switch (see above) and inserts a

simple reversible color transform into the processing

which significantly improves the compression.

Use this switch in combination with the -block N

switch (see above) for lossless JPEG coding.

CAUTION: A decoder with inverse color transform

support is required for this feature. Reversible

color transform support is not yet widely implemented,

so many decoders will be unable to view a reversible

color transformed JPEG file at all.

-bgycc Create big gamut YCC JPEG file.

In this type of encoding the color difference

components are quantized further by a factor of 2

compared to the normal Cb/Cr values, thus creating

space to allow larger color values with higher

saturation than the normal gamut limits to be encoded.

In order to compensate for the loss of color fidelity

compared to a normal YCC encoded file, the color

quantization tables can be adjusted accordingly.

For example, cjpeg -bgycc -quality 80,90 will give

similar results as cjpeg -quality 80.

CAUTION: For correct decompression a decoder with big

gamut YCC support (JFIF version 2) is required.

An old decoder may or may not display a big gamut YCC

encoded JPEG file, depending on JFIF version check

and corresponding warning/error configuration.

In case of a granted decompression the old decoder

will display the image with half saturated colors.

-dct int Use integer DCT method (default).

-dct fast Use fast integer DCT (less accurate).

-dct float Use floating-point DCT method.

The float method is very slightly more accurate than

the int method, but is much slower unless your machine

has very fast floating-point hardware. Also note that

results of the floating-point method may vary slightly

across machines, while the integer methods should give

the same results everywhere. The fast integer method

is much less accurate than the other two.

-nosmooth Don't use high-quality downsampling.

-restart N Emit a JPEG restart marker every N MCU rows, or every

N MCU blocks if "B" is attached to the number.

-restart 0 (the default) means no restart markers.

-smooth N Smooth the input image to eliminate dithering noise.

N, ranging from 1 to 100, indicates the strength of

smoothing. 0 (the default) means no smoothing.

-maxmemory N Set limit for amount of memory to use in processing

large images. Value is in thousands of bytes, or

millions of bytes if "M" is attached to the number.

For example, -max 4m selects 4000000 bytes. If more

space is needed, temporary files will be used.

-verbose Enable debug printout. More -v's give more printout.

or -debug Also, version information is printed at startup.

The -restart option inserts extra markers that allow a JPEG decoder to

resynchronize after a transmission error. Without restart markers, any damage

to a compressed file will usually ruin the image from the point of the error

to the end of the image; with restart markers, the damage is usually confined

to the portion of the image up to the next restart marker. Of course, the

restart markers occupy extra space. We recommend -restart 1 for images that

will be transmitted across unreliable networks such as Usenet.

The -smooth option filters the input to eliminate fine-scale noise. This is

often useful when converting dithered images to JPEG: a moderate smoothing

factor of 10 to 50 gets rid of dithering patterns in the input file, resulting

in a smaller JPEG file and a better-looking image. Too large a smoothing

factor will visibly blur the image, however.

Switches for wizards:

-baseline Force baseline-compatible quantization tables to be

generated. This clamps quantization values to 8 bits

even at low quality settings. (This switch is poorly

named, since it does not ensure that the output is

actually baseline JPEG. For example, you can use

-baseline and -progressive together.)

-qtables file Use the quantization tables given in the specified

text file.

-qslots N[,...] Select which quantization table to use for each color

component.

-sample HxV[,...] Set JPEG sampling factors for each color component.

-scans file Use the scan script given in the specified text file.

The "wizard" switches are intended for experimentation with JPEG. If you

don't know what you are doing, DON'T USE THEM. These switches are documented

further in the file wizard.txt.

DJPEG DETAILS

The basic command line switches for djpeg are:

-colors N Reduce image to at most N colors. This reduces the

or -quantize N number of colors used in the output image, so that it

can be displayed on a colormapped display or stored in

a colormapped file format. For example, if you have

an 8-bit display, you'd need to reduce to 256 or fewer

colors. (-colors is the recommended name, -quantize

is provided only for backwards compatibility.)

-fast Select recommended processing options for fast, low

quality output. (The default options are chosen for

highest quality output.) Currently, this is equivalent

to "-dct fast -nosmooth -onepass -dither ordered".

-grayscale Force grayscale output even if JPEG file is color.

Useful for viewing on monochrome displays; also,

djpeg runs noticeably faster in this mode.

-rgb Force RGB output even if JPEG file is grayscale.

This is provided to support applications that don't

want to cope with grayscale as a separate case.

-scale M/N Scale the output image by a factor M/N. Currently

supported scale factors are M/N with all M from 1 to

16, where N is the source DCT size, which is 8 for

baseline JPEG. If the /N part is omitted, then M

specifies the DCT scaled size to be applied on the

given input. For baseline JPEG this is equivalent to

M/8 scaling, since the source DCT size for baseline

JPEG is 8. Scaling is handy if the image is larger

than your screen; also, djpeg runs much faster when

scaling down the output.

-bmp Select BMP output format (Windows flavor). 8-bit

colormapped format is emitted if -colors or -grayscale

is specified, or if the JPEG file is grayscale;

otherwise, 24-bit full-color format is emitted.

-gif Select GIF output format. Since GIF does not support

more than 256 colors, -colors 256 is assumed (unless

you specify a smaller number of colors). If you

specify -fast, the default number of colors is 216.

-os2 Select BMP output format (OS/2 1.x flavor). 8-bit

colormapped format is emitted if -colors or -grayscale

is specified, or if the JPEG file is grayscale;

otherwise, 24-bit full-color format is emitted.

-pnm Select PBMPLUS (PPM/PGM) output format (this is the

default format). PGM is emitted if the JPEG file is

grayscale or if -grayscale is specified; otherwise

PPM is emitted.

-rle Select RLE output format. (Requires URT library.)

-targa Select Targa output format. Grayscale format is

emitted if the JPEG file is grayscale or if

-grayscale is specified; otherwise, colormapped format

is emitted if -colors is specified; otherwise, 24-bit

full-color format is emitted.

Switches for advanced users:

-dct int Use integer DCT method (default).

-dct fast Use fast integer DCT (less accurate).

-dct float Use floating-point DCT method.

The float method is very slightly more accurate than

the int method, but is much slower unless your machine

has very fast floating-point hardware. Also note that

results of the floating-point method may vary slightly

across machines, while the integer methods should give

the same results everywhere. The fast integer method

is much less accurate than the other two.

-dither fs Use Floyd-Steinberg dithering in color quantization.

-dither ordered Use ordered dithering in color quantization.

-dither none Do not use dithering in color quantization.

By default, Floyd-Steinberg dithering is applied when

quantizing colors; this is slow but usually produces

the best results. Ordered dither is a compromise

between speed and quality; no dithering is fast but

usually looks awful. Note that these switches have

no effect unless color quantization is being done.

Ordered dither is only available in -onepass mode.

-map FILE Quantize to the colors used in the specified image

file. This is useful for producing multiple files

with identical color maps, or for forcing a predefined

set of colors to be used. The FILE must be a GIF

or PPM file. This option overrides -colors and

-onepass.

-nosmooth Don't use high-quality upsampling.

-onepass Use one-pass instead of two-pass color quantization.

The one-pass method is faster and needs less memory,

but it produces a lower-quality image. -onepass is

ignored unless you also say -colors N. Also,

the one-pass method is always used for grayscale

output (the two-pass method is no improvement then).

-maxmemory N Set limit for amount of memory to use in processing

large images. Value is in thousands of bytes, or

millions of bytes if "M" is attached to the number.

For example, -max 4m selects 4000000 bytes. If more

space is needed, temporary files will be used.

-verbose Enable debug printout. More -v's give more printout.

or -debug Also, version information is printed at startup.

HINTS FOR CJPEG

Color GIF files are not the ideal input for JPEG; JPEG is really intended for

compressing full-color (24-bit) images. In particular, don't try to convert

cartoons, line drawings, and other images that have only a few distinct

colors. GIF works great on these, JPEG does not. If you want to convert a

GIF to JPEG, you should experiment with cjpeg's -quality and -smooth options

to get a satisfactory conversion. -smooth 10 or so is often helpful.

Avoid running an image through a series of JPEG compression/decompression

cycles. Image quality loss will accumulate; after ten or so cycles the image

may be noticeably worse than it was after one cycle. It's best to use a

lossless format while manipulating an image, then convert to JPEG format when

you are ready to file the image away.

The -optimize option to cjpeg is worth using when you are making a "final"

version for posting or archiving. It's also a win when you are using low

quality settings to make very small JPEG files; the percentage improvement

is often a lot more than it is on larger files. (At present, -optimize

mode is always selected when generating progressive JPEG files.)

GIF input files are no longer supported, to avoid the Unisys LZW patent

(now expired).

(Conversion of GIF files to JPEG is usually a bad idea anyway.)

HINTS FOR DJPEG

To get a quick preview of an image, use the -grayscale and/or -scale switches.

"-grayscale -scale 1/8" is the fastest case.

Several options are available that trade off image quality to gain speed.

"-fast" turns on the recommended settings.

"-dct fast" and/or "-nosmooth" gain speed at a small sacrifice in quality.

When producing a color-quantized image, "-onepass -dither ordered" is fast but

much lower quality than the default behavior. "-dither none" may give

acceptable results in two-pass mode, but is seldom tolerable in one-pass mode.

If you are fortunate enough to have very fast floating point hardware,

"-dct float" may be even faster than "-dct fast". But on most machines

"-dct float" is slower than "-dct int"; in this case it is not worth using,

because its theoretical accuracy advantage is too small to be significant

in practice.

Two-pass color quantization requires a good deal of memory; on MS-DOS machines

it may run out of memory even with -maxmemory 0. In that case you can still

decompress, with some loss of image quality, by specifying -onepass for

one-pass quantization.

To avoid the Unisys LZW patent (now expired), djpeg produces uncompressed GIF

files. These are larger than they should be, but are readable by standard GIF

decoders.

HINTS FOR BOTH PROGRAMS

If more space is needed than will fit in the available main memory (as

determined by -maxmemory), temporary files will be used. (MS-DOS versions

will try to get extended or expanded memory first.) The temporary files are

often rather large: in typical cases they occupy three bytes per pixel, for

example 3\*800\*600 = 1.44Mb for an 800x600 image. If you don't have enough

free disk space, leave out -progressive and -optimize (for cjpeg) or specify

-onepass (for djpeg).

On MS-DOS, the temporary files are created in the directory named by the TMP

or TEMP environment variable, or in the current directory if neither of those

exist. Amiga implementations put the temp files in the directory named by

JPEGTMP:, so be sure to assign JPEGTMP: to a disk partition with adequate free

space.

The default memory usage limit (-maxmemory) is set when the software is

compiled. If you get an "insufficient memory" error, try specifying a smaller

-maxmemory value, even -maxmemory 0 to use the absolute minimum space. You

may want to recompile with a smaller default value if this happens often.

On machines that have "environment" variables, you can define the environment

variable JPEGMEM to set the default memory limit. The value is specified as

described for the -maxmemory switch. JPEGMEM overrides the default value

specified when the program was compiled, and itself is overridden by an

explicit -maxmemory switch.

On MS-DOS machines, -maxmemory is the amount of main (conventional) memory to

use. (Extended or expanded memory is also used if available.) Most

DOS-specific versions of this software do their own memory space estimation

and do not need you to specify -maxmemory.

JPEGTRAN

jpegtran performs various useful transformations of JPEG files.

It can translate the coded representation from one variant of JPEG to another,

for example from baseline JPEG to progressive JPEG or vice versa. It can also

perform some rearrangements of the image data, for example turning an image

from landscape to portrait format by rotation. For EXIF files and JPEG files

containing Exif data, you may prefer to use exiftran instead.

jpegtran works by rearranging the compressed data (DCT coefficients), without

ever fully decoding the image. Therefore, its transformations are lossless:

there is no image degradation at all, which would not be true if you used

djpeg followed by cjpeg to accomplish the same conversion. But by the same

token, jpegtran cannot perform lossy operations such as changing the image

quality. However, while the image data is losslessly transformed, metadata

can be removed. See the -copy option for specifics.

jpegtran uses a command line syntax similar to cjpeg or djpeg.

On Unix-like systems, you say:

jpegtran [switches] [inputfile] >outputfile

On most non-Unix systems, you say:

jpegtran [switches] inputfile outputfile

where both the input and output files are JPEG files.

To specify the coded JPEG representation used in the output file,

jpegtran accepts a subset of the switches recognized by cjpeg:

-optimize Perform optimization of entropy encoding parameters.

-progressive Create progressive JPEG file.

-arithmetic Use arithmetic coding.

-restart N Emit a JPEG restart marker every N MCU rows, or every

N MCU blocks if "B" is attached to the number.

-scans file Use the scan script given in the specified text file.

See the previous discussion of cjpeg for more details about these switches.

If you specify none of these switches, you get a plain baseline-JPEG output

file. The quality setting and so forth are determined by the input file.

The image can be losslessly transformed by giving one of these switches:

-flip horizontal Mirror image horizontally (left-right).

-flip vertical Mirror image vertically (top-bottom).

-rotate 90 Rotate image 90 degrees clockwise.

-rotate 180 Rotate image 180 degrees.

-rotate 270 Rotate image 270 degrees clockwise (or 90 ccw).

-transpose Transpose image (across UL-to-LR axis).

-transverse Transverse transpose (across UR-to-LL axis).

The transpose transformation has no restrictions regarding image dimensions.

The other transformations operate rather oddly if the image dimensions are not

a multiple of the iMCU size (usually 8 or 16 pixels), because they can only

transform complete blocks of DCT coefficient data in the desired way.

jpegtran's default behavior when transforming an odd-size image is designed

to preserve exact reversibility and mathematical consistency of the

transformation set. As stated, transpose is able to flip the entire image

area. Horizontal mirroring leaves any partial iMCU column at the right edge

untouched, but is able to flip all rows of the image. Similarly, vertical

mirroring leaves any partial iMCU row at the bottom edge untouched, but is

able to flip all columns. The other transforms can be built up as sequences

of transpose and flip operations; for consistency, their actions on edge

pixels are defined to be the same as the end result of the corresponding

transpose-and-flip sequence.

For practical use, you may prefer to discard any untransformable edge pixels

rather than having a strange-looking strip along the right and/or bottom edges

of a transformed image. To do this, add the -trim switch:

-trim Drop non-transformable edge blocks.

Obviously, a transformation with -trim is not reversible, so strictly speaking

jpegtran with this switch is not lossless. Also, the expected mathematical

equivalences between the transformations no longer hold. For example,

"-rot 270 -trim" trims only the bottom edge, but "-rot 90 -trim" followed by

"-rot 180 -trim" trims both edges.

If you are only interested in perfect transformation, add the -perfect switch:

-perfect Fails with an error if the transformation is not

perfect.

For example you may want to do

jpegtran -rot 90 -perfect foo.jpg || djpeg foo.jpg | pnmflip -r90 | cjpeg

to do a perfect rotation if available or an approximated one if not.

We also offer a lossless-crop option, which discards data outside a given

image region but losslessly preserves what is inside. Like the rotate and

flip transforms, lossless crop is restricted by the current JPEG format: the

upper left corner of the selected region must fall on an iMCU boundary. If

this does not hold for the given crop parameters, we silently move the upper

left corner up and/or left to make it so, simultaneously increasing the

region dimensions to keep the lower right crop corner unchanged. (Thus, the

output image covers at least the requested region, but may cover more.)

The adjustment of the region dimensions may be optionally disabled by

attaching an 'f' character ("force") to the width or height number.

The image can be losslessly cropped by giving the switch:

-crop WxH+X+Y Crop to a rectangular subarea of width W, height H

starting at point X,Y.

A complementary lossless-wipe option is provided to discard (gray out) data

inside a given image region while losslessly preserving what is outside:

-wipe WxH+X+Y Wipe (gray out) a rectangular subarea of

width W, height H starting at point X,Y.

Other not-strictly-lossless transformation switches are:

-grayscale Force grayscale output.

This option discards the chrominance channels if the input image is YCbCr

(ie, a standard color JPEG), resulting in a grayscale JPEG file. The

luminance channel is preserved exactly, so this is a better method of reducing

to grayscale than decompression, conversion, and recompression. This switch

is particularly handy for fixing a monochrome picture that was mistakenly

encoded as a color JPEG. (In such a case, the space savings from getting rid

of the near-empty chroma channels won't be large; but the decoding time for

a grayscale JPEG is substantially less than that for a color JPEG.)

-scale M/N Scale the output image by a factor M/N.

Currently supported scale factors are M/N with all M from 1 to 16, where N is

the source DCT size, which is 8 for baseline JPEG. If the /N part is omitted,

then M specifies the DCT scaled size to be applied on the given input. For

baseline JPEG this is equivalent to M/8 scaling, since the source DCT size

for baseline JPEG is 8. CAUTION: An implementation of the JPEG SmartScale

extension is required for this feature. SmartScale enabled JPEG is not yet

widely implemented, so many decoders will be unable to view a SmartScale

extended JPEG file at all.

jpegtran also recognizes these switches that control what to do with "extra"

markers, such as comment blocks:

-copy none Copy no extra markers from source file.

This setting suppresses all comments

and other metadata in the source file.

-copy comments Copy only comment markers.

This setting copies comments from the source file,

but discards any other metadata.

-copy all Copy all extra markers. This setting preserves

metadata found in the source file, such as JFIF

thumbnails, Exif data, and Photoshop settings.

In some files these extra markers can be sizable.

Note that this option will copy thumbnails as-is;

they will not be transformed.

The default behavior is -copy comments. (Note: in IJG releases v6 and v6a,

jpegtran always did the equivalent of -copy none.)

Additional switches recognized by jpegtran are:

-outfile filename

-maxmemory N

-verbose

-debug

These work the same as in cjpeg or djpeg.

THE COMMENT UTILITIES

The JPEG standard allows "comment" (COM) blocks to occur within a JPEG file.

Although the standard doesn't actually define what COM blocks are for, they

are widely used to hold user-supplied text strings. This lets you add

annotations, titles, index terms, etc to your JPEG files, and later retrieve

them as text. COM blocks do not interfere with the image stored in the JPEG

file. The maximum size of a COM block is 64K, but you can have as many of

them as you like in one JPEG file.

We provide two utility programs to display COM block contents and add COM

blocks to a JPEG file.

rdjpgcom searches a JPEG file and prints the contents of any COM blocks on

standard output. The command line syntax is

rdjpgcom [-raw] [-verbose] [inputfilename]

The switch "-raw" (or just "-r") causes rdjpgcom to also output non-printable

characters in comments, which are normally escaped for security reasons.

The switch "-verbose" (or just "-v") causes rdjpgcom to also display the JPEG

image dimensions. If you omit the input file name from the command line,

the JPEG file is read from standard input. (This may not work on some

operating systems, if binary data can't be read from stdin.)

wrjpgcom adds a COM block, containing text you provide, to a JPEG file.

Ordinarily, the COM block is added after any existing COM blocks, but you

can delete the old COM blocks if you wish. wrjpgcom produces a new JPEG

file; it does not modify the input file. DO NOT try to overwrite the input

file by directing wrjpgcom's output back into it; on most systems this will

just destroy your file.

The command line syntax for wrjpgcom is similar to cjpeg's. On Unix-like

systems, it is

wrjpgcom [switches] [inputfilename]

The output file is written to standard output. The input file comes from

the named file, or from standard input if no input file is named.

On most non-Unix systems, the syntax is

wrjpgcom [switches] inputfilename outputfilename

where both input and output file names must be given explicitly.

wrjpgcom understands three switches:

-replace Delete any existing COM blocks from the file.

-comment "Comment text" Supply new COM text on command line.

-cfile name Read text for new COM block from named file.

(Switch names can be abbreviated.) If you have only one line of comment text

to add, you can provide it on the command line with -comment. The comment

text must be surrounded with quotes so that it is treated as a single

argument. Longer comments can be read from a text file.

If you give neither -comment nor -cfile, then wrjpgcom will read the comment

text from standard input. (In this case an input image file name MUST be

supplied, so that the source JPEG file comes from somewhere else.) You can

enter multiple lines, up to 64KB worth. Type an end-of-file indicator

(usually control-D or control-Z) to terminate the comment text entry.

wrjpgcom will not add a COM block if the provided comment string is empty.

Therefore -replace -comment "" can be used to delete all COM blocks from a

file.

These utility programs do not depend on the IJG JPEG library. In

particular, the source code for rdjpgcom is intended as an illustration of

the minimum amount of code required to parse a JPEG file header correctly.