# JPEGView

version 3.2

JPEGView was — for the most part — written, conceived, produced, and documented by Aaron Giles.

Behind the scenes, however, were quite a number of other rather important people you should know about.

## Dirty Work

First I want to thank all those other programmers who were nice enough to do some of the dirty work for me.   Their efforts in particular are responsible for making JPEGView work so smoothly:

Troy Gaul wrote Infinity Windoid, the code that draws the way cool floating windows everyone likes so much.

Ramon Felciano wrote Mercutio, the code which draws the menus and handles lots of weird keypresses.

The Independent JPEG Group, headed by Tom Lane, wrote the original color reduction algorithm, which was used as the basis for JPEGView's color reduction.

Ross Williams wrote LZRW1, the data compression technique used to squeeze the on-line help down to a reasonable size.

## Prerelease Hell

A big part of putting together a new release of JPEGView is the serious amount of testing that is needed to make sure everything works on all (most?) machines.   To this end, I would like to acknowledge all the alpha and beta testers who have contributed toward testing this JPEGView release:

Bob Andrews
Jim Brunner
Justin Collins
Shawn Connelly
John Coolidge
Ramon Felciano
Mike Fetzer
Troy Gaul
Stephen Haase

Paul Jacoby
Paul Jensen
Dave Johnson
Hugh Johnson
David Johnston
Mark Krueger
Tom Lane
Richard Lim
Tim Lowe
Robert Mah
Tom McDougal
Sam Neal
Tim North
Victor Norton
Brian Olson
Dan Pendergrass
Steve Poole
Charlie Reiman
Axel M. Roest
Eric Sack
Francois Schiettecatte
Bart Sears
James N. Stricherz
Steve Unger
Chris Webster

## Friends in High Places

A few special words of thanks are in order for several Apple employees who have made all the difference in helping me make JPEGView a success.

Mark Krueger was the guy who originally noticed JPEGView and helped me change it from a really basic hack into a robust QuickTime application.

Tim Lowe has been instrumental in helping me adopt and adapt to new Apple technologies — especially AppleScript — faster than I would have ever been able to without him.

And finally, thanks to Jim Pelkey and Linda Haas, for organizing the PowerPC developer's kitchen, and for making it such a great success.

## Others Who Have Made a Difference

Tom Lane wrote the second-largest help chapter, General JPEG Questions & Answers, and deserves some additional praise for founding and heading the Independent JPEG Group, whose free source code has made the widespread use of JPEG images possible.

John Coolidge, I recently discovered, is responsible for sorting and maintaining the burgeoning JPEG image archives at wuarchive.wustl.edu.   Here's hoping he can find the time (or the help!) to finally get all those pictures sorted!

More than 600 registered JPEGView users have so far taken the time to send in their postcards and letters, and believe me, the response has kept me going — keep 'em coming!

Finally, I want to thank my fiancée Shanti, for being understanding and even supportive of my programming addiction/obsession.

## Some Gory Details

JPEGView 3.2 consists of over 25,000 lines of vanilla C code, plus about 2700 lines of 68020 assembly for GIF decoding, color reduction, and direct-to-screen drawing.

The 680x0 version of JPEGView was compiled under MPW C and MPW Assembler. Debugging was usually done with THINK C 6.0, for familiarity's sake.

The PowerPC version of JPEGView was compiled under Apple's MPW PPCC compiler — but not very often. And I'm finally beginning to see the virtues of a two-machine debugger. I didn't even consider using assembly for this version.

The JPEGView JFIF Preview extension is about 400 lines of C code, and was compiled under MPW as well. It is a "fat resource" and runs natively on both platforms.

All the example AppleScript scripts were written and compiled using Apple's Script Editor, under AppleScript 1.0.

Development was done for the most part on a Macintosh IIcx 8/240 (yes, that's right folks, a 16MHz 68030), driving an Apple Portrait Display.

A lot of testing and all final builds were done on a Quadra 840av 16/330, with an Apple 16" display.

PowerPC testing was done on a PowerMac 8100/80 8/240, with an Apple 14" display.