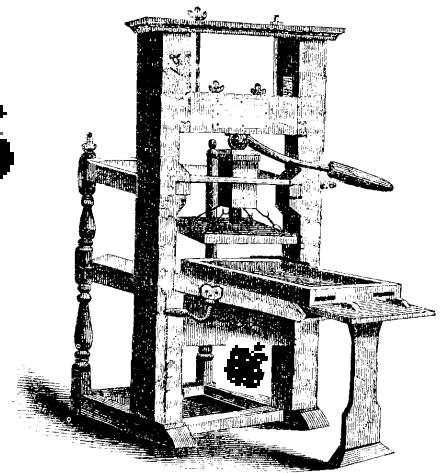


AppleSiders PRESS

2000

Ongoing Support For People Who Enjoy Using Their Computers.

Vol. 20
No. 2



AppleSiders Meeting NEW LOCATION!

Wednesday, February 16th, at
Northern Kentucky University's (NKU's) Band Room
126 Fine Arts Bldg.

Beginners Macintosh Q and A at 7:00 PM
AppleSiders Meeting will start at 7:30 PM

This meeting is open to the public.

A new year and a new meeting location! Come to the February meeting and check out the new meeting room.

The February meeting will feature the keynote address by Steve Jobs from MacWorld 2000 and a report by our own AppleSiders members who have attended MacWorld in person.

Plan to be at the meeting and give us your input on the new place.

Presented by Nick MacConnel and Larry Klug



Email Attachment Formats Explained by Adam C. Engst <ace@tidbits.com>

"What's the best encoding to use when sending a file to a Windows user via email?" I'll get to the correct answer shortly, but first let me explain the confusion.

Terminology

As I explained in "Macintosh Internet File Format Primer," in TidBITS-445_, there are usually two actions that take place for Macintosh files to be transferred via email: binary packaging and transfer encoding. Binary packaging, which is the realm of formats like AppleDouble, AppleSingle, and MacBinary, deals with the problem of other platforms not understanding that Macintosh files can have both data and resource forks. Transfer encoding, which is done via Base64 or uuencode, takes an 8-bit file and converts it to 7-bit ASCII text that can survive the journey through Internet email, which only *guarantees* safe passage for 7-bit ASCII data. The BinHex format combines both binary packaging and transfer encoding.

<<http://db.tidbits.com/getbits.acgi?bart=05066>>

Most email programs, including Eudora, EMailer, and Outlook Express, call the process of formatting an attached file for trans-

mission "encoding," thus conflating the binary packaging step with the transfer encoding step. That's not generally a problem for users, but caused some confusion in our quiz for people who know that Base64 (which garnered the most responses) is a transfer encoding format, whereas AppleDouble (the runner-up) is technically only a binary packaging format.

Now, you might be wondering, "So if AppleDouble is a binary packaging format, how does it survive being sent in email?" The answer is that an attachment, when packaged with AppleDouble and sent via email, is *also* automatically encoded via Base64. Under most circumstances, that Base64 encoding is transparent to users on both ends. Let me explain more about each email attachment format in turn.

The Correct Answer: AppleDouble

AppleDouble is the best format to use when sending attachments to Macintosh or Windows users, and particularly if you're sending the same file to Mac and Windows users at the same time. That's because AppleDouble breaks the Macintosh data and resource forks

of a file apart, then attaches them separately to the message, at which point they're also Base64 encoded for transfer. When a modern Macintosh email program sees the two attachments, it decodes the Base64 and then reassembles them into a single file. When an email program on another platform sees the two attachments, it throws the resource fork away and decodes only the data fork, since other platforms don't understand Macintosh resource forks.

Some have questioned my recommendation of AppleDouble over uuencode for sending files to Windows users. In reality, uuencode will work most of the time, but here's why I recommend AppleDouble over uuencode:

- If you're sending a file to both Mac and Windows users, and that file contains a resource fork that's useful but not necessary (as can be true of MS Word or Nisus Writer files), AppleDouble preserves the resource fork for Macintosh users while allowing Windows email programs to strip it transparently. In contrast, uuencode can cause headaches for Macintosh recipients because it removes the resource fork completely.

2000 EXECUTIVE COMMITTEE

- President** Gary Johnston
president@applesiders.com
- Vice President** Cyndi McDaniel
vp@applesiders.com
- Secretary** Barry Austern
secretary@applesiders.com
- Treasurer** H. Gerald Hoxby
treasurer@applesiders.com
- Parliamentarian** Larry Klug
larry@applesiders.com
- Newsletter Editor** Terry Crooker
editor@applesiders.com or
tcrooker@eos.net
- Membership** Al May
membership@applesiders.com
- Sysop** Gary Johnston
sysop@applesiders.com
- Pgm. Coordinator** Nick MacConnel
program@applesiders.com
- Needed** publicity
applevention



Notify the Membership Secretary of any changes in your mailing address or any problems receiving your newsletter. Annual dues are \$30.00.
Send renewal fees to:
 AppleSiders of Cincinnati, Attention Membership Secretary, 5984 Cheviot Road, Cincinnati, OH 45247
 Moving? Let us know. The Post Office does not forward standard mail.

Editor / Publisher—Terry Crooker
 Rapid Copy Printing, 5984 Cheviot Rd.
 Cincinnati, OH 45247
 741-4325—Home • 385-0888—Work
 editor@applesiders.com

AppleSiders PRESS is published monthly and sent via 3rd class mail to all full AppleSider members. In addition we exchange newsletters with other Apple User Groups across the country.

The AppleSiders PRESS is the official newsletter of the AppleSiders of Cincinnati. All contents are copyrighted, however all Apple Users Groups may reproduce articles without prior consent by giving credit to both the newsletter and the author and sending a copy of the publication to the AppleSiders mailing address in which the re-print appears.

“AppleSiders PRESS is an independent publication not affiliated or otherwise associated with or sponsored or sanctioned by Apple Computer, Inc. The opinions, statements, positions and views stated herein are those of the author (s) or publisher and are not intended to be the opinions, statements, positions or views of Apple Computer, Inc.”

Apple®, Mac®, and Macintosh™ are trademarks of Apple Computer, Inc. The AppleSiders PRESS is created with a PowerMac 7300/180 and Apple iMac computers and a HP LaserJet 5000N, using Ready, Set, Go 7.1™ 7.

- **Base64**, which is the transfer encoding format used by AppleDouble, is almost totally transparent to users, whereas uuencode is only usually transparent.

- **Uuencode** is not and will never be standardized, which leaves the door open for possible decoding problems.

- **Finally, AppleDouble** is the official Internet standard for sending Macintosh files in email, and there’s nothing new about it. To quote from the MacMIME specification (RFC 1740), from December of 1994:

“AppleDouble is the preferred format for a Macintosh file that is to be included in an Internet mail message, because it provides recipients with Macintosh computers the entire document, including icons and other Macintosh specific information, while other users easily can extract the data fork (the actual data) as it is separated from the AppleDouble encoding.”

<<http://www.cis.ohio-state.edu/htbin/rfc/rfc1740.html>>

Usage of and adherence to standards is a good thing, and it’s the main reason we have the Internet today. The sooner we eliminate obsolete email programs that understand only old attachment formats, the better off we’ll all be, and the sooner articles like this will be unnecessary.

Note that some Macintosh email programs don’t use the term “AppleDouble,” but instead call it “MIME,” and even Eudora has moved to calling it “AppleDouble (“MIME”).” That’s not entirely unreasonable, since “AppleDouble” doesn’t lead you to believe that it’s useful for sending files to Windows users. The problem, however, is that MIME will mean something different in Windows programs (and in Outlook Express 5.0 for the Mac), since Windows has no need for binary packaging at all, so “MIME” just means “Base64 encoding” for binary files that are attached to email.

The main problem with AppleDouble is that it doesn’t work in a few rare cases, mostly for people behind old email gateways that don’t properly support Internet standards. In that case, you’ll want to use whatever you can figure out that works.

AppleSingle

Whereas AppleDouble splits the data and resource fork into two files for attachment, AppleSingle bundles them together into a single file (much like MacBinary, which isn’t used for email). AppleSingle’s utility for email attachments is minimal, with spotty support through the Macintosh email universe. The main reason AppleSingle is still around is that the MacMIME specification also says that Mac files that lack a data fork, should be sent as AppleSingle. Overall, though, AppleSingle isn’t generally relevant to most people today.

uuencode

Again, uuencode is a transfer encoding format that throws away any resource fork information in the Macintosh file and converts the 8-bit file into 7-bit ASCII text. Although this destructive behavior with regard to the resource fork sounds bad, it actually isn’t quite as awful as I’ve made out because no Windows applications would be able to read the resource fork of a Macintosh file, even if it was transferred. If a Macintosh document *required* its resource fork, it wouldn’t be readable in Windows anyway. And if you were to send a Macintosh application via email with uuencode encoding, the application would be destroyed, but since a Macintosh application can’t run under Windows, there’s no loss.

The main reason uuencode still exists is historical—it was so prevalent in the days before MIME that it remains a necessary fallback when sending to folks using Windows and other platforms who haven't upgraded their email programs in several years.

Base64

As a transfer encoding format, Base64 is quite similar to uuencode but has the advantage of being modern and standardized. As with uuencode, if you send a Macintosh file using Base64 encoding, you'll lose the resource fork. Remember that Base64 is automatically used with AppleDouble, and if you're looking at a Windows email program, "MIME" in the context of attachment formats probably means Base64 encoding

<<http://www.cis.ohio-state.edu/htbin/rfc/rfc2045.html>>

I haven't quite been able to complete testing with AOL, but it appears that Base64 is the best choice for sending attachments to AOL users. Other formats work, but not as seamlessly. We'll have results next week.

BinHex

As I noted above, BinHex is an amalgam of binary packaging format and transfer encoding format. It combines both forks of a Macintosh file, then turns that 8-bit file into 7-bit ASCII text for sending. BinHex remains popular for sending files via email between Macintosh users, since essentially all Macintosh email programs understand BinHex. However, BinHex works far less well when sending files to Windows users, since few Windows email programs other than Eudora can decode BinHex. Many Mac users still rely entirely on BinHex for sending files via email, and if it works for you, that's fine. However, I would encourage you to switch to AppleDouble if possible, and here's why:

- Although BinHex is standardized, unlike uuencode, it's still an old format that would be good to relegate to the back burner for occasional needs so as to encourage everyone to upgrade to and support AppleDouble (most modern email programs do).

- If you send a file to both Macintosh and Windows users, there's a good chance that the Windows users won't be able to decode the file if you used BinHex.

- As discussed in "Calling Developers to MacBinary III" in TidBITS-444 BinHex doesn't support the new icon badges or custom routing information that's been available since Mac OS 8.5 shipped. That information may not be crucial, but there's no reason to use encoding formats that don't support it.

<<http://db.tidbits.com/getbits.acgi?tbart=05050>>

On the other side of the argument, BinHex includes an integral checksum at the end of the file, which means it's easier for an email program to check the attachment for corrup-

tion when BinHex is used than with other formats.

Quoted-Printable

Although quoted-printable, which most people have only seen via the QP button in Eudora, is a transfer encoding format, it's used not for attachments, but for high ASCII characters (special characters with diacritical marks, for instance). Since they aren't part of 7-bit (or low) ASCII, quoted-printable encodes such characters as an equal sign followed by a number. Since all other low ASCII characters remain intact, the message remains mostly human-readable despite the quoted-printable encoding.

An email message with equal signs at the ends of the lines is almost certainly a quoted-printable message that hasn't been decoded, usually because the Content-Transfer-Encoding header that specifies the quoted-printable encoding is missing. This problem crops up primarily in mailing list digests, since mailing list software removes most headers from messages before combining them into a digest. The main solution to this problem is MIME digests, which maintain headers for each message within the digest, facilitating bursting the digest into a mailbox and retaining special headers that specify transfer encoding.

The Role of Compression

Just when you thought you had a handle on all this, I'm going to throw in another variable: compression. It's often a good idea to compress files before sending them via email. That's especially true if you're sending large files or if you want to attach many files to a single message, since compressing the files into a single archive will save space and may make them easier to work with on the other end. I say "may" because one of the developers of Mulberry, which is primarily an IMAP email program, noted that in the IMAP mentality everything lives on the server until requested by the client, so being able to pick and choose one of several attachments to download is an advantage.

Compression can offer another advantage in that it may allow Macintosh files to survive transfer encoding that would normally destroy a file's resource fork. For instance, a StuffIt 5 archive stores everything in the data fork, so uuencode and Base64 won't damage it. Previous versions of the StuffIt file format could store some data in the resource fork (believe me on this, the information comes straight from the developers); moving everything to the data fork for cross-platform support was one of the main reasons Aladdin introduced the StuffIt 5 format. So, if you're using an email program that can compress files automatically before sending, uuencode and Base64 may work much better for you.

Of course, if you send a compressed file to a Windows user via email, they may be able to decode the attachment fine but find them-

selves left with a compressed file that they can't expand. If you anticipate sending that person lots of compressed files, encourage them to download a free copy of Aladdin Expander for Windows, which can decode a wide variety of formats, much like StuffIt Expander on the Mac side. On the other hand, if you need to send a compressed file to a Windows user only occasionally, both Drop-Stuff 5.5 and StuffIt Deluxe 5.5 can create Windows self-extracting applications (.exe instead of .sea) that your recipient could launch to expand.

<<http://www.aladdinsys.com/>>

Another use for compressing files is if you want a Windows or Unix machine to be a go-between for two Macs with files that require resource forks. By compressing the file down to just a data fork, you can ensure that no information is lost when the file stops temporarily on an intermediate machine before moving on to the destination Mac (via a network, floppy disk, or some other transfer mechanism). Of course, this assumes StuffIt Expander is available on the destination Mac.

What You Attach

Throughout this discussion, I've barely skimmed the surface of the types of files you're attaching to email. It won't do your recipient any good if you choose the proper attachment format if they can't open the document inside. The variables involved with choosing the proper format are numerous, but follow these recommendations and you'll be on the right track.

- If possible, ask what programs your recipient has that could open the file you're planning on sending. Then save the document in a format that you know can be opened by the recipient's programs.

- Make sure to give your files appropriate filename extensions for Windows, since without a .doc extension, for instance, Windows can't figure out that a file should open in Microsoft Word. Some email programs help with this; Outlook Express can optionally add extensions automatically, and Eudora goes to some effort to provide the extension in the meta-data about the attachment, without changing the actual name of the file.

- Most document formats for major productivity applications like the Microsoft Office suite, Adobe products, and so on, are fully cross-platform. If you stick to well-known applications that have Macintosh and Windows versions, conversion isn't likely to be a major issue. With files like spreadsheets and databases, sticking with the formats used by popular applications is your best bet, since the few available interchange formats are often quite limited and annoying to use.

- For word processing documents, Microsoft Word format (without the Fast Save option turned on) is the usually best choice and the most likely source format, followed by RTF



AppleSiders of Cincinnati
5984 Cheviot Road
Cincinnati, OH 45247

BULK RATE
U.S POSTAGE
PAID
CINCINNATI, OHIO
PERMIT NO. 5095

if either of you aren't using Microsoft Word. If all else fails, drop down to straight text, which you can send within the body of a message instead of as an attachment. As a matter of manners, if you're sending a document that's meant to be read-only, and the contents are pure text, just paste them into the body of the message rather than using an attachment.

- For graphics, stick either to the native file formats of cross- platform applications like Adobe Photoshop or to standard and well-supported graphics formats like GIF, JPEG, TIFF, EPS, and PNG.
- For sounds, stick to standardized formats like AIFF, MP3, and WAV.
- When in doubt, keep a copy of DataViz's MacLink Plus around (apart from a brief hiatus recently, it has shipped with the Mac OS for a long time) for translating between different file formats.

Sealing the Package

I hope this article has thrown some light on what has traditionally been a murky subject. The question that prompted the entire topic—the best format to use when mailing files to Windows users - seems relatively simple, and in an ideal world, we wouldn't even think about it. That's in large part why I recommend AppleDouble (with its transparent Base64 encoding) for everything—it *should* work with all modern email programs that adhere to Internet standards. The fact that it doesn't always work doesn't mean that we should rely on other formats for more than the occasional file to a user of an old email program. Rather, it means that we should work all the harder to make sure AppleDouble is supported everywhere so the entire question can fade into the depths of Internet lore, where it belongs. No one should even have to think about attachment formats, and only with full compliance with the MIME standards of AppleDouble and Base64 can we reach that point.

**Visit the AppleSiders Award
Winning Web Site**
<<http://www.applesiders.com>>

Odds & Ends

NEW Farallon HOMELINE ETHERNET & USB ADAPTERS provide connectivity for Macs, PCs & DSL/Cable modems with Ethernet or USB to share a single Internet connection, multi-player games, printers & files. <<http://www.farallon.com/tidbits/homeline>>

Aladdin Systems: **New Stuffit Deluxe 5.5 Now Shipping!** Faster Compression & Expansion, More File Formats! Send Self- Extracting Archives and Zip Archives to PCs! **SPECIAL DISCOUNT OFFER AT:** <<http://www.aladdinsys.com/deluxe/tidbitsoffer.html>>

DiskWarrior 2.0

Alsoft has introduced DiskWarrior 2.0, the latest version of its data recovery and directory optimization tool. In addition to its lauded directory optimization, fully functional preview feature, and near-magical CD-ROM that can start up a wide range of Mac systems (see "Fighting Corruption with Alsoft's DiskWarrior" in TidBITS-486), DiskWarrior 2.0 adds DiskShield, a new feature that checks the validity of any directory information being written to or read from your disks. In theory, DiskShield could prevent directory damage from occurring in the first place and alert you to potential problems before they grow into catastrophes. DiskWarrior 2.0 also includes the capability to graph fragmentation in disk directories so you can get an idea how much an optimized directory might help. DiskWarrior still lacks traditional disk optimization and brute-force data recovery offered by other disk recovery products, but the DiskWarrior CD ships with Alsoft's PlusOptimizer, a disk defragmenter that can be used on Mac OS Extended Format (HFS Plus) volumes. DiskWarrior 2.0 is \$70 plus shipping; currently the download-only version remains at 1.1. Upgrades from previous versions of DiskWarrior are \$30 plus shipping. [GD]

Aladdin IntelliNews:

The News You Need, When You Need It!
Now there's no need to bounce from Web site to Web site to find the info you need - IntelliNews does it for you! Check it out now at: <<http://www.digitalriver.com/aladdin/intellinews/22830>>