A2-Central™

formerly
Open-Apple

March 1992
Vol.8, No.2

ISSN 0885-4017
newstand price: $3.00
photocopy charge per page: $0.20

A journal and exchange of Apple II discoveries

Finder Fever

by Andy Nicholas

As I sit here typing this, System 6.0 for the Apple IIgs is in imminent danger of going Golden Master. That's when we take all of the development version numbers out of the components and promise not to touch the code any more (at least for this version). We've spent a lot of time working on System 6 and I've spent many hours tweaking the Finder so that people will enjoy using their Apple IIgs's.

Tim Swihart wrote last month's ***A2-Central*** article about the many features found in System 6. This month's article is about the many changes undergone by the System 6 Finder, Finder 6.0. The version number of the Finder changed from v1.3 for System 5 to 6.0 for System 6 to make the distinction clear that Finder 6.0 is meant for System 6.

First, Finder 6.0 has a new About box. The About the Finder menu item now displays a modeless (floating) dialog showing the version number of the Finder, version number of the System Software, and memory usage. Memory usage is broken down into the total memory in your Apple IIgs, the amount of memory currently available and the amounts used by the Finder, by the System (this includes the size of your ram disk), by any setup files (found in your system.setup folder), and by any desk accessories. The new About box is very useful for finding out just how much memory those unused DA's and Inits are actually taking up. If you open a window and wait about 15 seconds, you'll see the About box's memory usage figures being automatically updated.

Next in the Apple menu, you'll find a vastly changed help system. The new help system provides a single modeless dialog with a pop-up menu containing a list of available topics. Choosing a topic from the menu fills in the text edit field directly below the pop-up. Be sure to read all of the help (not just the boring stuff) because this is where we put the information on the goodies found in the Finder. That's right, even if you forget a hint or tip for the new Finder, you can always find it again without leaving the Finder (or fumbling through a pile of documentation).

To the right of the Edit menu you'll see the Windows menu that Tim talked about last month. When you open a window, the title of the window shows up in the Windows menu. When no Finder windows are open, you'll find a dimmed No Finder Windows menu item beneath the Stack Windows menu item. Stack Windows piles the windows on the desktop in order from bottom-most to top-most from left to right. Choosing an open window from the Windows menu brings it to the front of the other windows, something that's really handy if you open a bunch of windows, causing some windows to become completely obscured. And, if you just want to close a window without bringing it to the front, simply choose a window from the Windows menu while holding down the Option key.

Speaking of options, there are a whole bunch of keys that are useful when using the Finder. Most key combinations are easy to remember. For instance, holding down the Option key almost always means something related to closing. Holding down Option while clicking in the close box of a window (or when choosing the Close menu item) closes all the open windows. Holding down the Option key when double-clicking on a folder closes the window the folder is in after opening the window (this is called Tunneling). Holding down the Apple key while clicking on the title of a window displays a pop-up menu of the enclosing folders. Choosing a folder from the pop-up menu opens that folder's window, or brings it to the front if it's already open. Holding down the Option key while choosing from the pop-up menu also causes the window in whose title you are Apple-clicking to close (this is referred to as Reverse-Tunneling).

Holding down the Option key means close, usually. Option-Clean Up in the Special menu does "Clean Up by Name". Clean Up by Name is very powerful in that it alphabetizes your files and quickly places them on the Finder's clean up grid. Clean Up by Name on a window that is viewed by icon places the icons onto the large icon grid based on the width of the window. That is, if the window is only wide enough for 1 icon, you'll end up with a single column of icons, but if the window is wide enough for 5 icons, you'll have 5 icons across each row. Clean Up by Name on a window viewed by small icon places the icons into columns based on the height of the window. If you use the small icon view very much, Clean Up by Name will quickly become indispensable.

The Finder's regular Clean Up command is also much faster than Finder v1.3. And, as an added bonus, when you clean up a window, it stays cleaned up until you move the icons in the window, not just until you move the window's scroll bars. The ESC key deselects any selected icons. Pressing return while a single icon is highlighted gives that icon a rename field. Finder 6.0 even allows items on the desktop to be renamed!

Finder 6.0 fixes the "where did my files go?" syndrome. In Finder v1.3, when you drag an icon to a destination window, the Finder will get destination windows confused if the mouse is over the info bar, or over the scroll bars, or the grow box, or the title. If a folder

"WE'VE LET SOME OF THE 7TH GRADERS USE THE OFFICE COMPUTER AND IT LOOKS LIKE ONE OF THEM CRASHED INTO THE MONITOR."

resides directly beneath, say the info bar, and you drop what you're dragging on that very spot, then the icons you're dragging are put by Finder v1.3 into the folder underneath the info bar! Dragging icons on top of a window's control (scroll bars, info bar, title bar, etc.) results in Finder 6.0 putting the files into the window on which they are dropped rather than into a deeper level if a folder was lurking beneath the window's control.

Finder 6.0 fixes an annoying user-interface problem found in Finder v1.3 called "the impossible drag." If you place a small window directly in front of a larger window and then click on an icon in the large window with the intent to drag an icon into the small window, as soon as you click in the large window, Finder v1.3 brings the large window in front of the small window, completely covering it! Finder 6.0 fixes this by allowing you to click in the large window and drag an icon into the smaller, front window without bringing the large window to the front (and covering it in the process).

The Finder now provides for *smooth-launching* of applications that have the desktop tools bit set in their auxtype (see the Apple Filetype Technote for $B3 files for more information). A smooth launch is when the super-hires screen stays on while the Finder launches an application instead of having the Finder switch to the text screen only to have the application switch back to the super-hires screen. Hypercard IIgs v1.1, Teach, Archiver, and the System 6 Installer all use smooth launching.

If you've ever opened a folder with many hundreds of items, you probably decided not to do THAT again. With Finder 1.3, opening a fat folder could take minutes to finish. Your fear should subside greatly under Finder 6.0. If a folder that doesn't contain a Finder.Data file is opened, Finder 6.0 opens the window many times faster than Finder v1.3 did. In a test case, opening a folder on a hard drive containing 150 items used to take approximately 29 seconds using Finder v1.3. Finder 6.0 opens the same folder in less than 3 seconds.

Opening a folder that does have a Finder.Data file is faster in Finder 6.0, but the wait might still be annoying. For this reason, if you hold down the Control key while opening a folder, the Finder ignores any Finder.Data file and opens the window faster. If, as many people have done, you try to open a window with 500 or more items, Finder 6.0 will note that opening the window might take a long time and will put up a dialog asking whether you really want to open the window. In the interest of speed, the Finder doesn't use or save a Finder.Data file when opening a window with 500 or more items.

The Finder's Preferences have also changed. In addition to the options found in Finder v1.3, there is also *When starting up, Check 5.25 drives.* When not checked, this option causes the Finder to ignore your 5.25 drives when the Finder starts up. This is an easy way to defeat the noisy racket made by 5.25 drives (that brrrraaap! sound) when there's no disk in the drive and the Finder is starting. The standard setting for this preference is to poll the 5.25 drives.

Also in Preferences are a series of check boxes referring to the Finder's list views (By Name, By Date, By Size, By Kind). The standard setting for the *Show Date, Show Size, Show Kind,* and *Show disk info in header* preferences is to show the specified information. Not checking *Show Date* in preferences will cause the Finder to remove that column of information from all of the open windows that aren't being viewed *By Date.* By the way, there are key equivalents for all of the options in Preferences (we believe that key equivalents are a good thing and the result is that the Finder and System 6 are littered with them). Checking *Show disk info in header* causes the list views to have a double-height info bar that shows the space free and used on your disk as well as the column headings for list views.

Now we come to one of the Finder's most interesting features, Icon Info. When *getting info* on a file, you'll notice that there are now three cards for the rolodex-style info window. The third card, Comment, holds a single button labeled *Add Comment* if the file doesn't have a resource fork. Clicking on Add Comment displays a warning that adding a comment makes the file unusable from ProDOS 8, which can't open files with resource forks. If you choose to add the comment, the Finder adds a resource fork to the file and displays a text edit box. The amount of text isn't limited to 199 characters as it is on the Macintosh. You can type all you want, but we suggest you keep your comments in the visible area since this text edit control doesn't have a scroll bar and your audience might not bother to try to scroll through your lengthy message. The Comment card works for files on network file servers also. Once a comment is saved from a Apple IIgs for a file on a file server, the comment will show up in the info window of any Macintosh using Get Info on that same file.

If a file contains an rVersion resource (file's name, usually a copyright message, and version number), the contents of that rVersion resource are displayed on the General card of the info window. This is nice for discovering which version of a program you have without launching the application. (Encourage the software authors you know to include rVersion resources so you can use this feature.)

That was a brief overview of most of the user-visible changes to the Finder. Now we'll take a look behind-the-scenes at some of the stuff that makes the Finder really cool.

Scrolling both vertically and horizontally in Finder 6.0 is *much* faster than Finder v1.3. The Finder achieves the speed increase by only drawing the icons that are visible in a window instead of drawing **all** the icons in a window and then relying on Quickdraw II to clip off those items which aren't visible. If you're a programmer and using the Apple IIgs's desktop tools and think window scrolling is slow, try only drawing what's visible (or, in toolbox jargon, the window's visRgn). This tip alone could make the difference between a program people will love or hate.

Just dragging items around in a window in the Finder is noticeably faster. My test case for this was opening a folder with 1030 items, selecting all the items then dragging them a small amount in the same window. Finder v1.3 used to take 9 1/2 **minutes** to build the region used to erase and redraw the window's contents when the icons were moved. Finder 6.0 does this in 1.15 **seconds**. The Finder does this by only erasing and redrawing the icons that are visible, or are going to be made visible by being moved. Granted, this is an extreme example, but it illustrates some of the great stuff waiting in Finder 6.0.

When closing a window, Finder v1.3 used to first close the window, then redraw all of the icons beneath the window, then draw the *whoosh* rectangles. The result of this was that the Finder appeared to be **slow** when closing a window. Now, Finder 6.0 closes the window, but only redraws the background without the icons before drawing the whoosh rectangles. Only once the whoosh effect is complete does the Finder redraw the icons in the window. This results in more continuous motion (from your point of view), and it makes the Finder appear to be faster even though it's all just an illusion. Those of you who have the Beta version of System 6 know that when zooming down into a folder, the Finder would momentarily *blink* the folder whose window was being closed. Even little disturbances like that are now gone.

After dragging a group of files to a different disk, Finder v1.3 used to sit and count all the files (and files within folders, etc.) to be copied without telling you what it was doing. So, much of the time the Finder would seem to hang while it counted. Well, not any more! In the interest of providing much more feedback than before, Finder 6.0 counts **up** when it counts the number of files to be copied or verified. Instead of slowly counting upwards one file at a time, the Finder adds the number of files in each nested folder that you're copying. This causes the Finder to seem to randomly increase the number of files to be copied — the effect is really just to keep you entertained. When copying files, if a duplicate file already exists at the destination, a much different "this file already exists" dialog appears. Ron Lichty, Dave Lyons, Lee Collings and I tried to design a much better dialog that could be read and understood without resorting to displaying four huge buttons. In fact, not only have almost **all** of the Finder's dialogs been redesigned, but about 80% of the Finder's text is stored in resources. This means that a large portion of the Finder can be customized or internationalized through the use of a resource editor. All of the Finder's menus and menu items are in resources. A few things, such as the Icon Info window controls, are not stored in resources.

Finder 6.0 has special low-memory handling routines. These routines detect when the Finder is extremely low on memory and attempt to provide a warning before you reach the stage at which the system is so low on memory that certain tools start to behave strangely. And, by the way, if you ever receive one of these warnings, the

dialogs suggest corrective courses of action to help you free up more memory instead of taking the user-hostile approach of just saying "You're low on memory! Too bad!"

One of the all-time, most-hated Finder v1.3 bugs was when you'd drag a disk onto another disk to add the contents of the first disk to a folder on the second. The Finder would copy everything just fine, but then it would put away all the second disk's items and close all its windows. Finder 6.0 fixes this and also avoids putting icons away and closing windows when moving or copying a folder to another folder.

If you use Put Away (Command-Y) while the icon of a removable-media device is selected, that icon *whooshes* to the trash (as if you had dragged it there), the disk ejects, and the disk's icon is removed from the desktop.

Finder 6.0 uses the new GS/OS System 6 call JudgeName. If you try to rename an icon with a syntactically bad name for the file system used by that disk, the Finder displays a dialog telling you exactly what the syntax rules are and even suggests a new name that fits those rules. For example, if I try to rename a file on a ProDOS disk to "more stuff", Finder 6.0 complains that "more stuff" is not a valid ProDOS name. Finder then suggests a similar name, "more.stuff", which is valid, while explaining that "ProDOS names must begin with a letter. They may be up to 15 characters long and may contain letters, digits and periods."

Finder 6.0 uses the new GS/OS System 6 call HandleDiskInsert. If the Finder can't use a disk you just inserted (perhaps you insert a Macintosh HFS disk before installing the HFS File System Translator (FST)), HandleDiskInsert tries to identify the disk's file system and displays a dialog explaining which FST to install to use the disk. When using Finder 6.0, you'll notice many such instances where the Finder or System try to be friendly and helpful.

The way the Finder uses icons has changed dramatically. Finder 6.0 can use both the old System 5-style icon files and a new icon format based on resources. The information the Finder uses to keep track of documents and applications is stored in a new resource, rBundle. The idea is simple: the first time you launch an application that has an rBundle, the Finder extracts the rBundle resource and the rIcon resources attached to the rBundle and places all of them into a Desktop file. Each volume can have its own invisible Desktop file in the Icons folder. When the Finder starts up, it loads the contents of each Desktop file on each volume after loading the old-style icons from that volume.

The old-style icon format allowed an icon to be displayed based on a file's name, filetype or auxtype. rBundle allows an icon to be displayed based on a more extensive range of criteria: filetype, auxtype, name, create date/time, modification date/time, the file's access bits (locked, not locked), its network access (if applicable), HFS filetype, HFS creator, the contents of the GS/OS Option List, and the file's total length. For instance, if you wanted to display an icon for text files named Stuff created before 5:24 on February 6th of 1992 and modified after February 7th of 1992, you could. To edit the rBundle in an application or the Finder's Desktop files will require a third-party resource editor capable of editing an rBundle resource. If and when someone writes one, be sure to let me know.

Finder 6.0 keeps its own icons in its resource fork as rIcon resources. Several third-party utilities exist to edit rIcon resources. Feel free to customize the icon images in Finder 6.0. There's also a special resource to customize the eight standard window positions, the standard folder background and outline color, the standard quit setting, the standard trashcan position, the standard trash window position, and the standard clipboard window position.

The Finder internally converts old-style icon files into the newer rBundle format. If the application had its document icons attached to the rBundle when it was last launched, the application's document icons will be displayed when appropriate. When you double-click on one of the document icons, the Finder remembers in which folder the application used to be and attempts to launch it. If the Finder can't find the application (because — perhaps — you moved it), then it displays a dialog saying that the application can't be found and would you like to *Locate* the application. Choosing the *Locate* button displays a Standard File dialog that allows you to link the document back to the application. The Finder launches the chosen application after the Desktop database is updated. Whenever you are in the Finder, the link between the document and the application is whatever you chose in the Standard File dialog.

For instance, if the Finder can't find the *AppleWorks GS* application when you double-clicked on a text file, but you have System 6's Teach application and want Teach to open the file instead of *AppleWorks GS*, you would *Locate* Teach. From that point onward whenever you double-click on a text file, the Finder would try to launch Teach. If you ever decided you wanted to change the link between text files and Teach, all you need to do is hold down the Option and Control keys when double-clicking on a text file. The familiar Standard File dialog is displayed, giving you the chance to link text files to a different application. Just holding down the Option key when double-clicking on a document allows you to run a new application without making the link stick.

Finally, we come to the most exciting part of Finder 6.0 for developers: Inter-Application Communication (IAC), or Finder Extensions. This is one small area where the Apple IIgs really has the edge on the Macintosh. Macintosh Finder 7.0 doesn't have an extension mechanism that can be changed from version to version without requiring the recompilation of Finder Extensions.

System 6 provides a simple IAC mechanism through two toolbox calls, SendRequest and AcceptRequests. The Finder uses these to broadcast what it is doing to any Finder extension that wants to listen. Finder extensions that do stuff for the Finder and the rest of the system (ie, they aren't Finder 6.0 specific) usually take the form of Init files placed in your System folder's System.Setup folder. Finder extensions that the Finder can start up and shut down just like Inits should reside in a FinderExtras folder in the same folder as the Finder. The trade off for keeping a Finder Extension in the FinderExtras folder is that it will only use memory when the Finder is running, but the Finder will take longer to start up because the extension needs to be loaded every time the Finder starts. If the Finder Extension resides in the System.Setup folder, less time is used when the Finder starts up, but the extension is always using memory even if the Finder isn't running.

While an extensive discussion of how and when to use SendRequest and AcceptRequests codes is beyond the scope of this article, I've listed below the names of the codes that the Finder uses followed by a brief description.

finderSaysHello

The Finder sends finderSaysHello late in its start up process every time the Finder is launched. In response to finderSaysHello, an extension may make tellFinderAddToExtras requests to install menu items into the Finder's Extras menu. It is reasonable to watch for finderSaysHello and finderSaysGoodbye notifications to keep track of whether the Finder is present.

finderSaysGoodbye

The Finder sends finderSaysGoodbye early in its shutdown process to inform extensions that the Finder is going away (for whatever reason). Extensions should remove all of their Extras menu items at this time.

finderSaysSelectionChanged

The Finder sends this whenever the set of selected icons may have changed. On receiving this notification, an extension can make the tellFinderGetSelectedIcons call to see what icons are now selected. Extensions that have menu items in the Extras menu may want to call EnableMItem or DisableMItem on those items at this point.

finderSaysMItemSelected

The Finder sends this whenever any of the normal menu items are selected. This message is sent to all Finder extensions. The Finder decides whether to continue with the menu selection based on whether any extension accepts the request and whether a flag is true.

finderSaysBeforeOpen & finderSaysOpenFailed

When the user opens a document or application icon, the Finder sends finderSaysBeforeOpen. If the request does not get handled, the Finder tries to find an appropriate application to launch for the document. If that doesn't work, the Finder sends finderSaysOpenFailed to give extensions a chance to handle the request knowing that no application was found.

finderSaysBeforeCopy

Before the Finder does a file copy or a ChangePath call to move a volume on a volume, for each file the Finder calls finderSaysBefore

Copy. A flag tells the Finder to continue with the copy or abort.

finderSaysIdle

Finder broadcasts finderSaysIdle immediately before calling TaskMaster in its main event loop.

finderSaysExtrasChosen

finderSaysExtrasChosen notifies extensions that the user selected an item from the Extras menu.

finderSaysBeforeRename

When the user is about to rename an icon, the Finder sends finderSaysRename. If the request does not get handled, the Finder tries to rename the icon. If the request is handled, the Finder uses the return parameter to determine whether to perform the rename or not. This call is intended as a simple stop-gap security measure to (for instance) keep someone from renaming something that really shouldn't be renamed (the system folder on a server in a school, for example).

finderSaysKeyHit

If the Finder isn't able to deal with a key press, it sends finderSaysKeyHit before returning to the Finder's event loop.

askFinderAreYouThere

If the Finder is present, it always accepts this request and returns no error. This allows a Finder extension to determine if the Finder is executing.

tellFinderOpenWindow & tellFinderCloseWindow

Opens or Closes the specified Finder window. The clipboard, trash, and about windows can also be specified.

tellFinderGetSelectedIcons

Retrieves the selected icons and optionally their positions in a window or on the desktop.

tellFinderSetSelectedIcons

Sets the selected icons in a window or on the desktop.

tellFinderLaunchThis

Tells the Finder to launch a specified application.

tellFinderShutDown

Tells the Finder to shut down, restart, or quit to the previous application.

tellFinderMItemSelected

Tells the Finder to execute a menu-based action, just as if the user had chosen the item from the menus.

tellFinderMatchFileToIcon

Asks the Finder to search its internal data structures from all the volumes that have been seen so far to look for an icon that matches the specified search criteria.

tellFinderAddBundle

Tells the Finder to examine the contents of an rBundle from the provided pathname and add the contents of the rBundle (including attached rIcon, and rVersion resources) to the Finder's internal data structures or to the Desktop file.

tellFinderAboutChange

Informs the Finder that the contents of a directory have changed. If there is an open window for this directory, the Finder re-reads the directory's contents and brings the window up to date.

tellFinderCheckDatabase

Asks the Finder to look up whether or not it knows about a specific rVersion from all of the desktop databases which have been read into memory so far.

tellFinderColorSelection

Applies a color selection to the selected icons.

tellFinderAddToExtras

The Finder adds the specified menu item to the Extras menu (adding the Extras menu to the menu bar first if it is not already there). The item is added at the bottom of the menu unless otherwise specified.

tellFinderRemoveFromExtras

The Finder removes the specified menu item from the Extras menu.

askFinderIdleHowLong

Returns the number of ticks (1/60ths of a second) the Finder has been idle.

tellFinderGetWindowIcons

Returns a list of all of the icons and optionally their positions within a window or on the desktop.

tellFinderGetWindowInfo

Returns information about the type of the window (whether it's a directory window, the trash window, the clipboard window, etc.), the window's view, its file system, title, and pathname.

tellFinderSpecialPreferences

Tells the Finder to set some special preferences for this execution of the Finder only. The preferences are not saved. The Finder must be re-told about the preferences each time it is run. In the future this will allow certain special behavior modifications to the Finder by third-party utilities. Right now this allows the Finder to enable dragging hard drive partitions to the trash, marking that partition as offline. There isn't currently a utility to bring partitions back online, so this feature is saved until a third-party decides to use it (possibly by providing a Control Panel to mark partitions as offline or online).

Whew! A good example of a Finder Extension is the EasyMount Finder extension which we ship with System 6. Normally when you want to put a file server volume online, you have to use the AppleShare control panel to find the zone the file server is in, find the file server, log on to the file server by typing your user name and password, and finally choose the server volume. The EasyMount Finder Extension streamlines this process.

When you have a file server icon on the desktop, EasyMount allows you to click on the server icon and choose Create Server Alias from the Extras menu (the Extras menu isn't displayed in the Finder unless a Finder Extension requests that it be displayed). When you launch an EasyMount document, EasyMount hears a broadcast from the Finder of finderSaysBeforeOpen when you double-click on the alias file. EasyMount recognizes that the file is an EasyMount document and tries to put the corresponding file server online. EasyMount then returns control to the Finder, accepting the request.

These and other wonderful changes are all included in Finder 6.0. By the way, did I ever mention that you can now put folders on the desktop? <grin>

*(Andy Nicholas' sole project since joining Apple Computer in November 1990 has been Finder 6.0. Before working at Apple, he concentrated on graduating from Moravian College in Bethlehem, PA with a Bachelor's degree in Computer Science and a minor in History, when he wasn't working, that is, on **ShrinkIt** or **GS-ShrinkIt.** When asked recently what he did for fun, he replied, "not work!" He likes gadgets and sports (participating, especially basketball and ping pong). He dedicates this column to Randy Brandt whose wrist was broken while they were playing basketball at the **1991 A2-Central Summer Conference.**)*

Failing Grades II

by Tom Weishaar

John C. Dvorak is a computer journalist who has been around a number of years. One of his current residences is the back page of *MacUser*, where, in the February 1992 issue, he moons the Apple II community.

Dvorak is shocked that the computer steering committee in his local school district wants to buy more Apple IIe computers. Computer Pundit Dvorak has been predicting that Apple would kill the Apple II within weeks for the last five or six years. The news that Apple is not only still making Apple IIs, but that his local school district wants to buy them, reduced him to baring his canines and growling.

The bulk of his column is a series of quotes from various magazines that purport to show that the U.S. educational system hasn't taken advantage of computer technology.

Dvorak then draws a bead on his conclusion—that it's the Apple II's fault. "Obviously, the Apple II won't get junked until educators figure out that they are apparently the only group in the country that isn't benefiting from computer technology. In business and industry, computers improve efficiency and save money. Schools should be striving for both goals."

Dvorak overlooks a couple of important points about the educational system. Business and industry are money-centered. Success is measured on balance sheets. Computers are good at saving money,

so business and industry have invested heavily in junking generation after generation of computer system in an attempt to save as much money as possible.

The education system, on the other hand, is student-centered. Success is measured by standardized tests. And computer technology just hasn't been as good at raising students' test scores as it has at saving industry's money. Some educators, in fact, have argued that Dvorak's beloved Macintoshes actually hurt students by emphasizing looks over content (Marcia Peoples Halio, "Can the Machine Maim the Message," *Academic Computing*, Jan 1990).

While I'll readily agree that the Apple II family isn't as sexy as the Macintosh family, it does have significant advantages for educators. Right out of the box an Apple IIe costs less than any Mac (and a lot less than any useful Mac), which means schools get more machines per dollar. Next, those teachers who are using computer technology in education are, for the most part, using Apple IIs. Forcing these teachers to switch to another computer wastes their expertise. But the most important virtue of the Apple II for education is its stability.

The Apple IIe was introduced in 1983, a year before the first Mac. Because of its open design, a 1983 Apple IIe can easily use today's SCSI hard drives, can print on today's PostScript laser printers, can share files with other Apple IIs — as well as Macs and PCs — using today's AppleTalk networks, and has been able to overlay video images from a VCR or videodisk player on its own 1983 display screen for a couple of years already (something Macs are just now, with a great deal of fanfare, beginning to do). A 1983 Apple IIe can be an upstanding citizen in any school's 1992 computer system. Every month, educational publishers release software that is written to run on those 1983 Apple IIes. Macintoshes and MS-DOS machines, on the other hand, have, by design, limited life spans. The original 128K Macintosh, introduced a year after the Apple IIe, doesn't even make a decent paperweight in 1992.

It's obvious that there will be Apple IIs in America's schools for years to come. Ask administrators who opted for the original 8088 IBM-PC in 1983 what kind of new software is being released for those computers this month. Ask administrators who opted for 128K and 512K Macs in the mid-80s whether the Apple IIgs wouldn't have been, in retrospect, a much better investment.

If you're not getting the message, let me turn conventional wisdom on its head—buying Macs or PCs is an immediate investment in obsolescence. Buying an Apple II is an investment in a computer that will be in America's schools almost as long as chalk. If you had as little money to spend as the U.S. education system does, would you invest in computers that you knew would be obsolete before your freshmen were seniors?

Dvorak reminds me of the "experts" who sent Peace Corp Volunteers to India in the early 1970s to teach the farmers of the Punjab how to use the new tractors they were buying. The experts were concerned that the only thing the farmers used the tractors for was pulling a wagon to town and back. Better send some American boys in there, the experts said, and show those villagers what tractors are for.

It only took us American boys only a few weeks to realize that what the Punjabi farmers needed tractors for was pulling a wagon to town and back. Transporting high-yielding crops to market with traditional methods was a major problem. Doing field work for high-yielding crops with traditional methods wasn't.

If Dvorak spent a few weeks in his kid's classroom, he'd quickly learn that computer technology of whatever scope isn't the answer to most of the classroom teacher's challenges. And he'd learn that what works for American industry doesn't necessarily work for American education, just as what works for the American farmer doesn't necessarily work for the Punjabi farmer.

The Apple II family provides the modest, stable level of computer technology that American education needs. The challenge is to get the country's computer "experts" off education's back. Let's let teachers teach with the tools they ask for and stop shoving expensive technology — that wastes their time without solving their problems — down their throats.

(Footnote One: Certain members of the Apple II community often complain that the Apple II unit inside Apple doesn't do enough for them. It's very interesting to look at this "lack of support" from a Mac user's perspective. Dvorak writes, "The fact is, the Apple II has to be expunged from schools and the Mac substituted. Apple itself was, until recently, of little help in making this happen. As one Apple spokesperson said to me, 'It used to be verboten to mention the Mac as an educational computer.' The reasons were twofold: members of the Apple II clique in Cupertino jealously clung to their education-market stronghold because they realized that it was going to be their last stand. At the same time, the marketing types who labored for years to make the Mac an acceptable business alternative to the DOS standard feared that the word educational would tarnish its reputation as a 'serious' computer." The Apple II crew inside Apple has worked and continues to work against incredible pressure to keep the computer teachers want alive. Let's appreciate their efforts.)

(Footnote Two: Dvorak is a journalist, so we can't really expect him to spend a few weeks in his kid's classroom. Instead, I recommend he investigate a scandal in the computer industry—a scandal that has gone completely unreported. In fiscal 1988 Apple Computer's Apple II family revenues were over $1 billion for the year. Product lines that generate $1 billion a year are rare. Just five years later, in fiscal 1993, Apple II family revenues will be less than a tenth as much. Products lines that fall from such heights so quickly are extremely rare. But what is unique is that Apple's executives collected big bonuses while doing nothing to rescue the product line. Nobody on earth is worth the kind of money Apple's president makes. This is one of the biggest scandals in American business history. If it happened in any other company, executive heads would roll in baskets instead of in cash.)

Miscellanea

Some of us around here were wondering whether Apple had quietly squashed the SuperDrive controller card that was announced at KansasFest last summer. It's a card for the Apple IIe and IIgs that allows those computers to use the 3.5 inch, 1.44 Megabyte Apple SuperDrive. It turns out that the card **does** exist and **has** existed for a number of months now. We didn't realize it because all Apple did was to replace its old 3.5 disk controller card with the new card on its product list. No name change, no press release that we could find. The official name of the SuperDrive card is "Apple II 3.5 Controller Card," part number A0076LL/A. The list price is $149.00. We strongly suspect that the new card will also work with the $189 AMR 3.5 drive that we carry, transforming it into a 1.44 drive, as well. We'll keep you posted.

An annoying HyperStudio 3.0 bug has been isolated by graphic/computer artist extraordinaire, Bo Monroe. Bo noticed that occasionally, HyperStudio would fail to save a stack to the 3.5 drive. The drive light would come on for a second but no information would be saved and no error message would appear on the screen. Bo called this bug annoying, I called it devastating. Quite by accident, as he was talking me through a change to the *Studio City* master disk, we discovered that the culprit was a volume name longer than 13 characters. The moral of this tale is limit your disk names to 13 characters if you use HyperStudio 3.0.

A number of people have called our offices in the last few weeks asking about the status of Chinook Technologies. We can give you some concrete information. Sequential Systems of Lafayette, Colorado is now selling and servicing the Chinook product line as well as honoring previous warranties. In the past, they have specialized in products aimed primarily at the educational market but carry a number of items of interest to the general Apple II community. ***Q-System GS+*** with AppleTalk allows non-networked computers access to networked printers. They also carry a 52 meg hard disk for Apple IIe, IIc , IIgs, or Laser 128 computers that connects to the existing Unidisk port, no SCSI card required. For more information contact Sequential Systems at 1200 Diamond Circle, Lafayette, CO 80027, orders: 800-759-4549, technical support and information: 303-666-4549. Chinook itself will now focus its energies on the development of software for the Apple II marketplace.

Those of you living in/around or visiting the Boston area on April 4 and 5th will probably want to set some time aside for the Apple Expo East. Co-sponsored by the Boston Computer Society and Event Specialists, the site of the Expo is the Park Plaza Castle, the same hotel that housed the very first AppleFest 12 years ago. For further information, call Event Specialists at 617-784-4531.

It has become apparent that there is a world-wide shortage of Apple Hi-speed SCSI cards. One supplier didn't expect his shipment until the first of March. CV Technologies reports a slight backorder situation on its increasing popular RamFAST SCSI card as well. Maybe Apple IIgs owners are getting the message that in order to fully utilize their computers a hard drive in not a luxury, it's a necessity!

The latest efforts of the Alliance International, Inc. involves a letter writing campaign aimed at the United States Postal System. They hope to get a stamp issued recognizing the Apple as the computer that started the Computer Revolution. Enthusiasts are urged to write to: Citizen's Advisory Committee, c/o General Manager, Stamps Division— USPS, Washington, DC 20265-6351. You can mention that you support the Alliance International's request to recognize the Apple computer with a stamp and elaborate if you wish.

***The AppleWorks Educator* has announced a special offer on its best selling AppleWorks Educator Teacher Resource Disk for *A2-Central* subscribers.** This double-sided 5.25 disk contains a multitude of files and templates tailored for students and teachers using AppleWorks. It normally sells for $14.45 but will be made available for $12.00 (Canadian/Mexican orders please add $1.50, foreign orders add $3.00 for shipping). For subscription information or to order the disk write to *The AppleWorks Educator*, P.O. Box 72-A2, Leetsdale, PA 15056.—edr

Ask (or tell) Uncle DOS

Easier answer? (not!)

In ***A2-Central*** (Vol. 8, No.1, *"Icons and inter-connections"* p. 8.8a) you stated "that an opportunity for some enterprising programmer would be to write a utility to translate files based on the "rich text" format."

Aw, come on, Dennis. Don't make it so complicated. <grin> *AppleWorks* already does this! (And why does Richard Katz need *TimeOut* to convert *AppleWorks* word processing files to a text file on disk?)

If I understand what you mean by "rich text", it is not a plain vanilla ASCII file, but one which also has all sorts of commands embedded within the text, such as for various fonts, boldfacing, etc. Right?

To do this from plain old AppleWorks, go to the "Printer Information" menu, select add a printer (hopefully there is room for one), add a custom printer, when asked what slot it's in, choose "7", "print onto disk". When setting up printer codes, there are (in addition to the CPI and boldfacing, etc.) six special codes. Of course you'll want more than six, so choose a sequence that often appears as the start of a code you'll be using (like "Esc ("). When typing the document, these special codes will insert whatever you want into your document, and for further codes that don't "fit", use the special code from the special code table, followed by actual typing in your document. When ready to print, **don't** choose "A text (ASCII) file on disk", but choose the custom printer you have set up. This will print a "rich text" file on disk, with all the commands you'd ever need, all nicely embedded in your ASCII text file!

One nice thing about this is that the document can also be printed on a printer, with boldfacing, etc. in the right places.

I print "rich text" ASCII files onto disk many times everyday, because I print my files through a mainframe VAX onto the office laser printer, all through "rich text" ASCII files made by AppleWorks.

Terrell Smith
Marburg, W. Germany

In the old days I used to say "assume I'm a genius", but I've been humbled many times since then.

One problem is you don't understand what I mean by "Rich Text Format", or RTF (that's okay, I only found out about it a few months ago). Maybe I should have elaborated. (The response was gleaned from an on-line reply I gave to a question on GEnie and I wasn't around at the editorial deadline last month to elaborate.)

Text with embedded printer codes isn't the same as Rich Text. The problem with the file you're creating from AppleWorks is that it still contains codes that will only work with a specific printer, e.g. 'Escape I' to start boldface printing on an ImageWriter II. Rich Text is trying to avoid computer- or printer-specific formats when transferring files.

Rich Text Format is an evolving format for expressing text attributes (size, style, formatting, etc.) in the form of a vanilla ASCII file that other programs can decide to support for importanting and exporting text. A few programs already do, such as ***Microsoft Works*** *for the Macintosh and* ***MacWrite II****.*

Here's a sample sentence that I entered into a Works word processor document:

Starting with plain 12-point Geneva, let's see what Rich Text does with **bold**, *italic*, superscript ***bold italic superscript***, and a switch to 14 point Courier.

Here's what the document looks like when saved in the Rich Text Format:

```
{\rtf0\mac {\fonttbl}
{\colortbl\red0\green0\blue0;\red255\green255\blue255;\re
d255\green0\blue0;\red0\green255\blue0;\red0\green0\blue2
55;\red0\green255\blue255;\red255\green0\blue255;\red255\
green255\blue0;}
\pgnstart1 \pard\ql\sl240\ri-720 \li0 \fi0
\plain\f3\fs24\cf0 Starting with plain 12-point Geneva,
let's see what Rich Text does with \plain\b\f3\fs24\cf0
bold\plain\f3\fs24\cf0 , \plain\i\f3\fs24\cf0
italic\plain\f3\fs24\cf0 , \plain\up6\f3\fs20\cf0 super-
script\plain\f3\fs24\cf0 , \plain\b\i\up6\f3\fs20\cf0
bold italic superscript\plain\f3\fs24\cf0,and
\plain\f22\fs28\cf0 a switch to 14 point
Courier\plain\f3\fs24\cf0 .\par}
```

When you import this gibberish into ***MacWrite****, it automagically (through the RTF translator included with* ***MacWrite****) converts it back into formatted text within your* ***MacWrite*** *document that matches the original we created in* ***Microsoft Works.***

It really isn't gibberish; some of the embedded formatting codes like "\plain" (for the "plain text" style) are fairly obvious. As the format is evolving, documentation of the various formatting codes is appearing. They're too long to reproduce in ***A2-Central,*** *but I've seen at least one explanatory document in the Borland area on GEnie. (page 765, file #2917) So, like I said, what we need is for a few enterprising souls to get out in the world, look this over, and write some translators for use on the Apple II.*

Now, about using ***TimeOut*** *to export the text: the reason I recommend* ***TimeOut PowerPack's AWP.TO.TXT*** *utility to export files is that I've never seen it add a spurious carriage return. I have seen* ***AppleWorks 3.0*** *do this; a GEnie user suggested that this will occur when AppleWorks paginates the document for printout (this happens even when you print to disk). If a paragraph is split between two pages, AppleWorks will insert a carriage return at the break point. All I know is that I was always getting a few spurious carriage returns using AppleWorks's "print to disk" commands and the problem went away when I started using AWP.To.TXT. (Trust me; we import and export text files a lot here and I'm continually fighting this problem.)*

Finally, defining a disk-based "printer" for saving print codes does have a very useful function: if you examine the resulting file with the right utility (one that lets you see control characters) it lets you diagnose whether the codes AppleWorks is sending are really proper for your printer. This isn't 100 per cent foolproof since AppleWorks doesn't send all its codes to the disk-based printer (for example, it doesn't send the "reset" sequences at the beginning of each line that it sends when it prints to a "real" printer). But if your boldface isn't coming out bold, this is an easy way to check and see if the boldface codes are actually being placed into the file.

The best diagnosis is to put your printer in a "hex dump" mode (for the ImageWriter II, you can do this by holding down the "select" button as you power on the printer). This can waste a lot of paper, so you want to get the test document as small as possible before you try the hex dump.—DJD

Fun Stuff

I enjoyed reading Henry Markham's letter from Scunthrope, England (January 1992, p7.95) in which he tweaks all of us, specifically me, to tell about what we have done.

In my computer class for the over-55's, I like to remind students to read their manuals now and then. We almost make a joke of the "M" word. To reinforce this, before class starts and while students are gathering, I project, among other items of the day, a picture of Winston Churchill on the big screen, then show words coming from his lips, "Have you read your Manual today?" It's nice that my group doesn't have to be told who that big bulldog face is and it makes the point painless. How did I do this with a mere Apple II?

The picture of Winston Churchill came on a DOS 3.3 disk as part of a RAM expansion I bought for my Apple II in 1982, called *Expand-a-RAM* by Prometheus Products, Inc. of Fremont, Calif. The disks included a demo program in which 5 or 6 pictures were stored on a ramdisk, then loaded in succession into the main computer ram quite rapidly, giving a vivid display of the speed of BLOADing a 34 block file from ram.

The presentation plan was to BLOAD Churchill, then after a few seconds wait, to bload a duplicate file, which contained the words coming out of Churchill's mouth. I got the words onto the duplicate picture using another 1982 program, *Alpha Plot* by Bert Kersey and Jack Cassidy, aka Beagle Bros. The program was slow and awkward, no mouse but with patience, the manual question was installed on the picture and saved to disk. The ten year old disk still works!

All this still as DOS 3.3. However, I am now working in my class with an Apple IIc Plus which has a built in 3.5 inch floppy and didn't want to bother getting the 3.5 to operate under DOS 3.3. I figured that I could use the RAMdisk automatically installed by ProDOS. I used *Copy II Plus* to copy the before and after picture files of Winston from the DOS 3.3 disk to ProDOS. Then I put a short BASIC program together which, on startup, loads each picture, then saves it to S3,D2, which is the RAMdisk. Then a rotating display program loads the pictures in turn, so Winston asks the question. It was a big hit!

By the way, here's an ethical question: If I send a copy of these 1982 programs, now "out of print" to Harry Markham, am I doing anything wrong?

Henry Linton
Wilmington, DE

The copyright on a work doesn't expire just because the work is not sold commercially. We've heard several people raise logical arguments for and against "free distribution" of these types of programs, but the portions of the law that are clear on the matter indicate that it's not okay to distribute a copyrighted program without the express permission of the copyright holder. Period. (Copyrights do expire if not renewed, but the terms are now so long that the chances are the computer will be ancient history by the time the program passes into the public domain.)

We've heard the arguments that you might be safe passing the program around since no one may care to enforce the copyright. Being sensitive about copyrights ourselves, we can't encourage this.

What we'd like to see is the copyright holders of these programs come forward and give some terms for distribution. Beagle Bros did this with many of their "classic" programs.—DJD

Floating slots

Can the Floating Point Engine be used in slot 3 of an Apple IIe when there is a RamWorks III board installed in the auxiliary slot? Slot 3 is the only slot that I have available.

Billy Boehme
Arlington, Texas

Some expansion cards contain programs that help to "drive" functions of the card; these programs are usually contained in ROM (read-only memory) that is assigned to a specified range of the IIe memory when the card is installed.

With a memory expansion card installed in the auxiliary slot of the IIe, a slot 3 card cannot use the ROM memory space normally assigned to it; that space is turned over to the driver programs for the 80-column text display that are contained in part of the IIe's internal ROM. That's why most manufacturers tell you not to install their peripheral cards (disk interfaces, printer interfaces, etc.) in slot 3.

The FPE happens to be an exception since it doesn't require the ROM space. All it needs is the address space assigned to the physical slot, which is still available even if the IIe auxiliary slot is in use. The FPE's software patches for AppleWorks and Applesoft can access the card by using the physical address space.—DJD

(So what Dennis is saying, Billy, is that , yes, you can stick the Floating Point Engine in slot 3. <grin>) —edr

Blankered out

Reference Joe Douglas' "Blank screen" on p 7.72 in the October 1991 issue. The following is a simple but effective way of implementing a screen blanker in Applesoft.

Blanker is a demonstration program that shows how to implement screen blanking in Applesoft programs that use text display for menus and controls. The program is quite simple and will run on any Apple computer.

The heart of Blanker is found in lines 180 to 230. As soon as the initial menu is set up (lines 80-120) the program starts counting. This is the reference used to decide if time is up and it's time to blank the screen. If a keypress is detected, the counter is reset and the keypress is checked for validity. If it is acceptable the program continues. If it is not, the program restarts the counter. Uninterrupted, the timer reaches the preset count, D. At this point, the screen is cleared and the program waits for a keypress. When a key is pressed, the display is restored and input is again accepted.

Because the program is written in Basic, it is affected by several factors. The speed of the computer affects how fast the counter counts. An Apple IIgs counts from zero to 1000 in just under seven seconds. An Apple IIe, IIc, or II Plus counts from 0 to 1000 in about 20 seconds. If the computer is running with active interrupts, the counter will "lose" time over a long period of time. These two considerations are not significant unless you are a stickler for accuracy. If you are looking for two minutes of delay before blank and you get 110 seconds, this isn't serious.

The demo program is written to use the 40 column display. The method demonstrated will also work with the 80 column display.

```
10 REM PROGRAM SCREEN BLANKER
20 REM VERN L. MASTEL
30 REM TIMINGS ARE FOR APPLE II, IIE, IIC or IIGS IN
   NORMAL SPPED
40 REM REFERENCE SPEED IS APPROX 20 SECONDS/THOUSANDS
50 REM IIGS IN FAST SPEED IS APPROX 7 SECONDS/THOUSAND
60 D=1000
70 DATA MENU CHOICE 1, MENU CHOICE 2, MENU CHOICE 3,
   MENU CHOICE 4, MENU CHOICE 5, MENU CHOICE 6
80 TEXT : NORMAL : POKE 34, 0: HOME : RESTORE :V = 6
90 PRINT : HTAB 7: PRINT "****SCREEN BLANKER****": PRINT
   : PRINT : GOSUB 320
100 FOR X = 1 TO 6: READ A$(X): HTAB 12: PRINT A$(X):
   NEXT X: GOSUB 320
110 PRINT : HTAB 7: PRINT "USE <- -> ARROWS TO CHOSE" :
   PRINT : HTAB 4: PRINT "PRESS RETURN TO ACCEPT SELEC-
   TION"
120 INVERSE : VTAB V: HTAB 12: PRINT A$(V - 5)
130 |K = PEEK (49152): IF K < 128 THEN GOTO 180: REM IF
   NO KEY GOTO TIMER LOOP
140 POKE 49168,0: NORMAL : VTAB V: HTAB 12: PRINT A$(V -
   5): IF K = 141 THEN 240
150 IF K = 136 OR K = 139 THEN V = V - 1: IF V = 5
   THEN V = 11
160 IF K = 149 OR K = 138 THEN V = V + 1: IF V = 12
   THEN V = 6
170 GOTO 120
180 X = X + 1: REM INITIALIZE COUNTER
190 IF PEEK (49152) > 128 THEN X = 0: GOTO 150: REM KEY
   PRESSED..RESET COUNTER..EXIT TO MENU
200 IF X < D THEN 180: REM  IF COUNT IS LESS THAN LIMIT
   KEEP COUNTING
210 NORMAL : HOME : REM  COUNT REACHED..CLEAR SCREEN
220 GET T$
230 X = 0: POKE 49168,0: GOTO 80: REM  KEY WAS
   PRESSED..RESUME PROGRAM
240 V = V - 5
250 IF V = 1 THEN  HOME : PRINT "SET SCREEN DELAY TO 20
   SECONDS": D = 1000: GOTO 310
260 IF V = 2 THEN  HOME : PRINT "SET SCREEN DELAY TO 40
   SECONDS":D = 2000: GOTO 310
270 IF V = 3 THEN HOME  : PRINT "SET SCREEN DELAY TO 80
   SECONDS" :D = 4000: GOTO 310
280 IF V = 4 THEN HOME  : PRINT "SET SCREEN DELAY TO 2
   MINUTES" :D = 6000: GOTO 310
290 IF V = 5 THEN HOME :  PRINT "SET SCREEN DELAY TO 4
   MINUTES" :D = 12000: GOTO 310
300 IF V = 6 THEN HOME :  PRINT "SET SCREEN DELAY TO 8
   MINUTES" :D = 24000
310 PRINT : PRINT "PRESS ANY KEY TO ACTIVATE & CONTIN-
   UE";: GET A$: GOTO 80
320 FOR L = 1 TO 39: PRINT "-";: NEXT : PRINT : RETURN
```

Vern Mastel
Mandan, ND

(Note that this program does work, however it won't save any data that is on your screen.) —edr

Basic training

I have an extremely aggravating problem in BASIC. I teach programming in high school and imagine I'm one of the few still using DOS 3.3. A number of years ago, a student had a program which contained hi-res graphics. The program started doing all kinds of weird things like producing extra code, illegal line #'s, etc. This problem has increased to the point where over half of the programs using hi-res written by my

students this year have experienced problems. I'm hoping that this is something simple. Did you know that B. Dalton's and other book stores no longer offer a single book on Apple BASIC programming?

The first indication of something wrong is when the graphics are being placed on the screen. A little graphic line appears flashes through different colors. When you list the program, there will be anywhere from one to hundreds of extra line numbers. They are not in the proper numerical order nor can they be deleted. The program itself sometimes will not save; the drive runs a long time and then the screen goes into inverse. Sometimes loading the program only gives you the 'Program too large' error. A few years ago I met someone who had heard of this and gave me a poke that does help sometimes although we still lose programs.

Richard Wilson
Winfield, Mo.

We're well aware that bookstores don't carry Apple II books anymore. But we do! Check out this month's catalog.

The problem with your students' programs is that the Apple II graphics screens are smack in the middle of the memory in which Applesoft programs reside. Applesoft gets to use memory from $0800 up to $BFFF (having DOS 3.3 or BASIC.System resident in memory reduces the upper limit to about $9600 or less depending on the number of buffers opened in memory). The first high-resolution page (the one initialized with Applesoft's HGR command) is at $2000 to $3FFF, and the second page (HGR2) is at $4000 to $5FFF.

Back in Volume 2 we published a number of articles on Applesoft memory usage that explain all this.

If your program is longer than $1800 bytes, when HGR is issued it will eat the end of your program (the memory cleared by HGR is the same memory where the end of your program resides). Also, since variables other than strings are normally allocated in the memory just following the end of your program, they'll get stomped on, too. Or your program can return the favor by creating new variables, which will be written into the screen memory and show up as "garbage" in the graphics display (there's a way this can happen with strings, too; more on that in a moment).

You can check the endpoint of your program (variables not included) by using this formula:

```
PRINT PEEK(105) + 256*(PEEK(106): REM pgm end
```

For those with only ten fingers, $2000 (the start of high-res page 1) is 16384 and $4000 (start of page 2) is 24576. Watch out when you exceed these values and you're using the corresponding memory for graphics.

The simple solution is: if your program is shorter than $1800 bytes, use the Applesoft LOMEM: command to push the variables above the memory used for the screen. For example:

```
10 LOMEM: 16384: REM start variable storage at $4000
```

will protect the high-resolution screen from variable encroachment (but not from a swelling program).

If your program is likely to grow beyond $1800 bytes in size, but not beyond $3800 bytes, then you may want to use HGR2 instead of HGR. Then use:

```
10 LOMEM: 24576: REM ante upped to $6000
```

to protect the screen from variables.

From here it gets trickier. If you're running the program from disk and both it and it's variables will fit in the memory from $4000 to (about) $9600 then you may want to relocate it as it's loaded from disk:

```
10 IF PEEK(104) < 96 THEN POKE 103,1:
        POKE 104,96: POKE 96*256,0:
        PRINT CHR$(4);"RUN THIS.PROGRAM":
        REM moves program to start at $4000
```

Now you can use $2000 to $3FFF for graphics, but $0800 to $1800 isn't much use for your Applesoft program (though it would now be available for assembly language programs). You'll want to fix these pointers before running your next Applesoft program; the easiest way is to exit to ProDOS via the quit command:

```
1000 PRINT CHR$(4);"BYE": REM unload BASIC.System
```

Beyond this, you get into segmenting your program to make it fit into the available space, which we covered back in Volume 2.

We haven't forgotten strings (text variables like "Your name here"). These are stored at the top of available Applesoft memory and work down. If you create many strings, they can stack up until they work down into the graphics screen memory from above, even if the program is very small. Applesoft will only clean up discarded strings when memory gets tight (and remember, Applesoft doesn't watch the memory used for graphics) or when you force "garbage collection" via a FRE(0) (or BASIC.System "FRE") statement. If you see garbage creeping into your screen while doing a lot of string manipulation, you'll need to use the same techniques as above to protect the graphics memory.—DJD

Tips from Texas II

When Beagle Bros suspended telephone support for its Apple II products last fall, I felt, as many others did, "if they won't support their products, then why should I?" I quickly changed my mind, however, because I found that support today is as good as before — you just have to know where to find it. Some suggestions:

— Be online. There are users on every service who are experts on many of the *TimeOut* applications. Many are even more knowledgeable in some areas (*UltraMacros, ReportWriter*) than Beagle's own techs. Since Beagle's Pro-Beagle bulletin board is a long-distance call, many of the regulars have stopped calling there. In my opinion, most users have settled on GEnie. Beagle's own techs and programmer Mark Munz are still on America Online (keyword BB).

— Know your Beagle Buddy. Many Beagle Buddies (myself, Kent Hayden, Ray Merlin, and several others) are in conference with Beagle Bros tech support **every week** at our own expense and are in touch in other ways as well. We relay questions and usually have an answer within 24 hours. Just because the answer may be second-hand doesn't make it any less viable or correct.

— *NAUG, TimeOut-Central, The AppleWorks Educator,* and *TI&IE* organizations all have experts who can provide telephone or letters-to-the editor help. It's probably best to call or write directly to the individual, and enclose return postage, since a general letter to the editors may get lost in a stack of paper and disks. If you don't send postage, you can hope for a reply in the next issue of the publication (probably two months away), but don't count on it. Space is at a premium, and your question may be equipment-specific or obscure.

— And, not to give myself yet another a plug, but be a Kingwood customer and a official subscriber to *Texas II* (not just a downloader!). Kingwood customers know that they can call and ask about any of the TimeOut products, *even* if they didn't buy it from Kingwood. I wouldn't think of "warning" a customer that no support is available, because it is understood that it is. We're in touch with programmers and techs at Beagle, JEM, and other AppleWorks developers every day.

Beagle's attitude is not cavalier. They have given outstanding support to the community for more years than most. Perhaps it's because of that unprecedented level of support that they are no longer able to provide enough personnel to cover the phones today, while they work to bring *BeagleWorks* (a Mac product) to the market. I do not hesitate to recommend that you buy their Apple II products. It's in all of our best interests — and up to us — to continue to support Apple II in a positive way.

Beverly Cadieux
2018 Oak Dew Drive
San Antonio, Texas 78232

We've received innumerable letters concerning the recent pull-back in support of Beagle Bros products. I found Bev's message on GEnie recently and thought it deserved publication.—edr

A2-Central™

[redacted]

Publisher: **Tom Weishaar**
Editor: **Ellen Rosenberg**

with help from:
Denise Cameron **Dennis Doms** **Sally Dwyer**
Dean Esmay **Greg Miles** **Jeff Neuer**
Jean Weishaar

A2-Central,—titled ***Open-Apple*** through January, 1989—has been published monthly since January 1985. World-wide prices (in U.S. dollars; airmail delivery included at no additional charge): $34 for 1 year; $60 for 2 years; $84 for 3 years. All back issues are currently available for $2 each; bound, indexed editions of our first six volumes are $14.95 each. Volumes end with the January issue; an index for the prior volume is included with the February issue.

The full text of each issue of ***A2-Central*** is available on 3.5 disks, along with a selection of the best new public domain and shareware files and programs, for $90 a year (newsletter and disk combined). Single disks are $10.

Please send all correspondence to:

A2-Central
P.O. Box 11250
Overland Park, Kansas 66207 U.S.A.

ISSN 0885-4017
Printed in the U.S.A.

GEnie mail: A2-CENTRAL
Voice: 913-469-6502
Fax: 913-469-6507