A2-Central™

formerly Open-Apple

May 1990
Vol.6, No.4

ISSN 0885-4017
newstand price: $2.50
photocopy charge per page: $0.15

A journal and exchange of Apple II discoveries

The shell game

Back when *ProDOS Inside and Out* was written, an appendix was included that was intended to help those who wanted to choose between the use of ProDOS or DOS 3.3 to make their decision. A few of the arguments are still valid, but time has taken us to a point where DOS 3.3 cannot be considered a viable option for Apple II users wishing to be anything other than anachronistic. (Unfortunately, those of us who make our arguments public have to live with them for a long time.) Three things have occurred since the time the book was written to alter the impact of the discussion in that *ProDOS Inside and Out* appendix.

First, Apple delivered AppleWorks, which became a major force as the integrated application that sold hundreds of thousands of Apple IIe and IIc computers. AppleWorks was a child of the ProDOS operating system; if you wanted to use AppleWorks, you had to use ProDOS.

Second, Apple released the UniDisk 3.5, the first mass-market removable media drive with a respectable storage capacity (800K versus the paltry 140K of standard Apple 5.25 drives) accepted by Apple II users. ProDOS is clumsy when limited to 140K disks, but becomes elegant on larger capacity devices. DOS 3.3 is just the reverse; acceptable on lower capacity disks but visibly out of its element on large volumes. The UniDisk 3.5 propelled a move to a "mass storage environment" that Apple's overpriced Profile hard disk never managed.

The third factor had nothing to do with Apple; this was the appearance of programs that made ProDOS easy to use for the rest of us. Apple's utilities and manuals were universally horrific for neophyte users. The single most important oversight was an explanation of how to exit one program and start another without re-booting each time. Apple's standard user manuals, although they do explain the concepts of files and directories, don't effectively explain the mechanics to their customers wishing to use ProDOS 8 (the IIgs Finder solves many of the problems if you are a IIgs user booting through GS/OS). For example, let's assume you have booted into AppleWorks and wish to quit and run a communications program to download a file. How do you get to that next program without rebooting?

The concept is one of a *program selector;* software that allows you to select the next program you wish to use. ProDOS 8 includes a mechanism that allows AppleWorks (or any program) to request ProDOS to execute a self-contained routine intended to allow the user to select the next program. Unfortunately, ProDOS's standard routine is intimidating in the process it uses. The first prompt a user meets is:

```
ENTER PREFIX, (PRESS "RETURN" TO ACCEPT)
```

and ProDOS expects you to provide a name that indicates the default directory you want ProDOS to use. Next, you'll be asked to:

```
ENTER PATHNAME OF NEXT APPLICATION
```

Now ProDOS expects you to type the filename (or pathname, if the file is in a directory other than the one specified by the prefix) of the next program you intend to run.

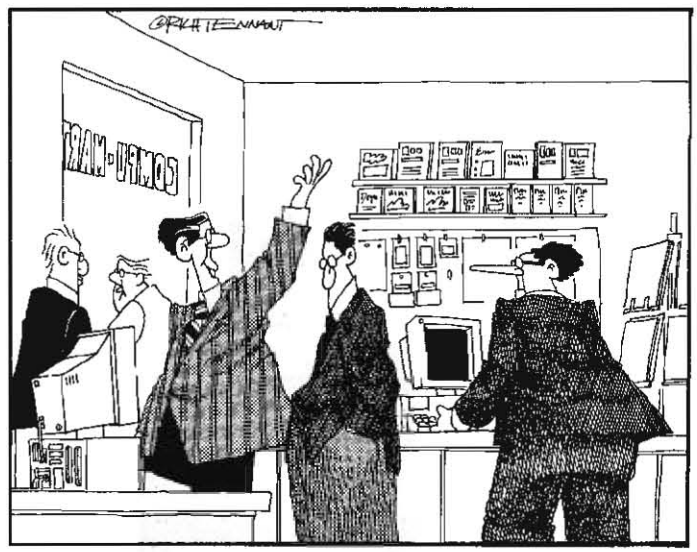
What's wrong with this system? Once you are looking at the prompt, ProDOS won't let you list the volume names or file names for any disks! The standard ProDOS selector doesn't accept slot and drive numbers or any command to list the contents of disks; you have to know where you're going before the ProDOS selector will let you get there.

Thankfully, Apple has documented the location of ProDOS's selection routine and the protocol for starting new programs in section 5.1.5 of the *ProDOS 8 Technical Reference Manual.* It is possible to replace the normal selector with a more useable program selector. The *Windows* utility on our monthly disk installs replacement selector code Karl Bunker added to the program (the assembly language source code is included in the STANDARD.TOOLS subdirectory as WIND.QUIT.S).

Program selectors come in three general styles with regard to how the user selects the next program: the use of a command line (you type the program's name), a menu-based system (the program's name is displayed as an element of a list), or a symbolic model (the program file is represented by an image).

A command line interface (we'll call it a "shell", to indicate that it serves as a protective shell around the raw internals of the disk operating system) expects you to memorize a few command "key words" that allow you tell the program what to do. The key words themselves vary from one shell to another, and are often cryptically abbreviated. For the MS-DOS shell interface, the command to list files within a directory is "DIR" (short for "directory"), the same command for a unix system is "ls" (short for "list"). By specifying a keyword along with a filename, you can perform operations on specific files; for example, in MS-DOS "ERASE THISFILE.TXT" tells MS-DOS to ERASE (delete) the file THISFILE.TXT from the disk.

One problem with using a shell as a standard user interface is that there can sometimes be hundreds of commands (and hundreds of filenames) to remember; commands to copy files, configure devices, and so on. Many users consider this to be confusing. But if we limit



"...AND TO ACCESS THE PROGRAMS 'HOT KEY', YOU JUST DEPRESS THESE ELEVEN KEYS SIMULTANEOUSLY. HERB OVER THERE HAS A KNACK FOR DOING THIS THAT I THINK YOU'LL ENJOY—HERB! GOT A MINUTE?"

our use of a shell to finding and executing a program, the number of commands needed is often reduced to only a few. This is the case with the Apple II; after all, Apple does give you utility programs to do things like copy and delete files, so if you know how to run the program everything falls into place.

The classic Apple II shell involves the use of disk commands from BASIC. This interface was built into DOS 3.3 itself, but for ProDOS it requires the use of a specific program represented by the BASIC.System file found on Apple's ProDOS 8 system disks. When you boot ProDOS, it looks for the first file of type "SYS" (the filetype is shown when you list the contents of a ProDOS disk or directory with most utilities) with a name ending in ".SYSTEM" and executes that program. Therefore, if you initialize a new ProDOS disk with your (ProDOS) system utilities and copy the files PRODOS and BASIC.System to the disk, you can boot that disk and initially use BASIC.System as your shell.

Selecting a disk. From BASIC.System, there are three commands that you need to know in order to execute most files. The first command, PREFIX, allows you to select which disk (or ProDOS directory) you will be dealing with, assuming you either know the name of the disk or the slot and drive where the disk is located.

If you know the disk is installed in a drive and you know the disk's name, using PREFIX followed by the name will tell BASIC.System you want that disk established as the one to use by default. For example, if you have inserted your AppleWorks master named /APPLEWORKS/, the command:

```
PREFIX /APPLEWORKS
```

will tell BASIC.System you want to deal with the /APPLEWORKS volume. (Notice that a ProDOS volume name always begins with a "/" character; a trailing "/" is optional.) If a disk with the name of /APPLEWORKS/ can't be located, BASIC.System will return a "PATH NOT FOUND" error. If the disk appears to be damaged, BASIC.System returns an "I/O ERROR" message ("I/O" is short for "input/output", indicating the transfer of data to or from the disk).

If you don't know the name of a volume but you do know the slot and drive assignment of your disk drive, you can let BASIC.System find the volume name and select that volume for use by inserting the disk into the drive and using two parameters set off by commas. The first, "Sn" (where "n" is the slot number), provides the slot number of the disk; the second, "Dn" (where "n" is the drive number), provides the drive number. For a ProDOS disk inserted in slot 6, drive 1 (the normal location for a 5.25" startup drive) the command would be:

```
PREFIX,S6,D1
```

Typing PREFIX without any parameters will display the current setting for the prefix. If there is no prefix currently specified, PREFIX (without parameters) will attempt to access the default disk (usually the last disk device accessed using slot and drive numbers; if the drive is empty, you'll get an I/O ERROR message) and display the name of any ProDOS volume found in that drive. This does not "select" the volume, so to set the prefix to the volume in the default device you need to use two commands:

```
PREFIX
```

to display the name and (continuing with the example of /APPLEWORKS/):

```
PREFIX /APPLEWORKS
```

(that is, PREFIX plus the name returned previously) to select the volume.

Once you've identified the name of a volume, it's a good idea to add the name to your disk label if it isn't already there. In the future, all you'll need to do is look at the label to confirm the disk's name.

Selecting a directory. The second command allows you to list the files on a disk or in a subdirectory, and that command is CATALOG. CATALOG produces an 80-colume listing, so BASIC.System also provides a form called CAT which produces a listing tailored to the 40-column screen.

CAT (or CATALOG) used by itself will attempt to list the contents of the last accessed disk or of the current directory. Assuming you have used PREFIX to select a specific disk, the current directory will be the root directory of that disk. For our AppleWorks 3.0 master (3.5) disk, for example, CAT produces the following list of files:

```
/APPLEWORKS

NAME              TYPE  BLOCKS  MODIFIED
PRODOS             SYS      32  22-MAR-89
APLWORKS.SYSTEM    SYS      26   3-AUG-89
SEG.00             BIN       9   3-AUG-89
SEG.XM             BIN       9   3-AUG-89
SEG.RM             BIN       9   3-AUG-89
SEG.AM             BIN       9   3-AUG-89
SEG.EL             BIN      12  27-DEC-88
SEG.PR             BIN       9   3-AUG-89
SEG.ER             BIN       6   3-AUG-89
SEG.AW             BIN      98   3-AUG-89
SEG.WP             BIN      89   3-AUG-89
SEG.DB             BIN      87   3-AUG-89
SEG.SS             BIN      76   3-AUG-89
MAIN.DICTIONARY    BIN     395  14-JUL-89
CUST.DICTIONARY    TXT       1  15-JUN-89
FASTCOPY           SYS      40  17-JUL-87
LETTER             AWP       3  26-JUN-89
MAILLIST           ADB       4  26-JUN-89
INCOME             ASP       4  26-JUN-89
TEXT               AWP       1  26-JUN-89
DATA               ADB       3  26-JUN-89
SPREAD             ASP       3  26-JUN-89
SAMPLE.FILES       DIR       2   2-AUG-89
BLOCKS FREE:  289     BLOCKS USED: 1311
```

Each file listing contains, among other items, two specific items we are interested in. One is the file name:

```
APLWORKS.SYSTEM SYS      26   3-AUG-89
```

and the other is the file type:

```
APLWORKS.SYSTEM SYS      26   3-AUG-89
```

The file name is used to identify the specific file we are interested in. For the purposes of trying to run a program, four types of files may be of interest; the CAT command displays these as "SYS" (system file), "BAS" (Applesoft BASIC program), "BIN" (binary file; this may contain an executable program, or it may not), and "TXT" (text file; a limited number of these known as "EXEC" files can be executed with BASIC.System). Notice that "SYS" and "BAS" files are usually executable programs (there may be occasional exceptions), but that "BIN" and "TXT" files most often are not executable programs (at least, it's best not to assume that they are programs). There is no 100% reliable way to assure that a file is an executable program, but the above rules serve as a practical guide.

There is a fifth file type that may be of interest; the "DIR" file. This is a subdirectory that may contain further sets of files. If you don't see the file you want to execute in the volume directory, you can use PREFIX followed by the subdirectory name to enter that subdirectory:

```
PREFIX SAMPLES
```

(Notice that the subdirectory name is not preceded by a "/"!) and then use CAT to list any files it contains. CAT (without a pathname) will only allow you to see the files for the directory that you have currently specified. To return to the volume directory for the disk, you again must type:

```
PREFIX /APPLEWORKS
```

(see, we left off the trailing "/" this time). This method of maneuvering through directories is tedious, but it has the advantage of being systematic. On the AppleWorks disk, you can use the sequence:

```
PREFIX /APPLEWORKS/
CAT
PREFIX SAMPLES
CAT
PREFIX ADVANCED.FILES
CAT
PREFIX /APPLEWORKS
```

to locate the "ADVANCED.FILES" directory within the "SAMPLES" directory on the "/APPLEWORKS/" volume and list the files it contains. Finding no executable files, you return to the volume directory.

There is a shortcut. If you know precisely where a subdirectory of interest is contained, you can combine the names of all the directories you pass through into one long "path name" to the subdirectory and feed it to the PREFIX or CAT command thus:

```
CAT /APPLEWORKS/SAMPLES/ADVANCED.FILES
```

Notice that one (and no more than one) "/" character is used to separate each directory name from the next, and that the order of the names (from left to right) is the same as that used in the individual PREFIX commands above (from top to bottom). There are other tricks, but we won't try to rewrite *ProDOS Inside and Out* here.

Selecting a program. There is no clear rule for identifying the program file you are looking for, other than to know from reading the program's documentation or by experience (that is, you tried it before and it ran consistently). One of the files you will notice on the AppleWorks disk is APLWORKS.SYSTEM, and that has a high probability of being a main program file because it is a "SYS" file and it follows the ProDOS ".SYSTEM" naming convention, indicating it is intended to startup on booting the AppleWorks disk. So, after using:

```
PREFIX /APPLEWORKS/
```

to select the directory, all we need to do is to try and execute the file. We do this using the BASIC.System "smart run" command, which is just a dash followed by the name of the file we want to execute:

```
-APLWORKS.SYSTEM
```

The disk should spin and we'll be in AppleWorks.

Selecting the proper prefix is a necessary step! The APLWORKS.SYSTEM file uses the value of the prefix to locate other files it needs to use (the files labelled with names starting with "SEG.", among others). It is not uncommon for programs to use the prefix in this manner.

Once you're convinced AppleWorks is running, you may want to quit and get back to your program selector. However, this won't be BASIC.System; as we mentioned earlier, standard ProDOS has its own selector. As you learn the prefix and the filename to enter for your disks, you can elect to use ProDOS's selector. Or, like most people, you may prefer to avoid the standard ProDOS selector as much as possible.

One alternative, if you've mastered launching programs from BASIC.System, is to make up a "selector disk" containing BASIC.System. You can then insert this disk and supply its name as the prefix and "BASIC.System" as the application. Then you'll be back to a "selector" with more options. To exit back to the selector installed in ProDOS, just type "BYE" from the BASIC prompt.

But even if you become familiar with BASIC.System, there are times where it won't be acceptable as a selector. Some programs are too large to execute directly while BASIC.System is in memory. And some people won't ever find using a shell acceptable to them. An option is to replace the ProDOS selector with something more palatable; alternative selectors are available from sources of public domain software or shareware (including on-line services and user-group libraries) and commercially.

Augmenting Apple's system. Several smaller selectors have been made available that fit completely in the same space as the standard ProDOS selector, such as *Squirt* ($8.50 from Synesis Systems, P.O. Box 1308, Gilbert, Ariz. 85234), and *Bird's Better Bye* (available on many Beagle Bros products, on our monthly disk, and on some other commercial products). These selectors are more of the "menuing" type. You select the disk volume you want to deal with (either from a list, or by "toggling" to each device). The selector displays a list of directories and SYS files on the chosen disk; you select the subdirectory, and then you select a "SYS" file to run. Most of these selectors, like ProDOS's built-in selector, will only execute files of type "SYS".

Another option is to replace ProDOS's selector with a small program that attempts to load a more complicated and capable selector. This has the disadvantage of requiring that a disk with the program for the larger selector be available when you return to the selector. On the old 140K floppy disks, this inconvenience may outweigh ease-of-use, but with larger disks (3.5 disks, RAM disks, or hard disks) the space required is negligible and the ease of use increases tremendously as the number of files on the volume increases. The selector can be any one of the three types mentioned.

The extended 8-bit selector that we had our first experience with (and primarily still use) is Glen Bredon's *ProSel*. It is based on a menuing system that allows the user to pre-define a set of applications with their location (by "prefix" pathname and file name) and even to pass the name of a file to applications that accept a "startup" path. A common example of a program which accepts a startup path is BASIC.System itself, which normally will attempt to execute a file with the name of STARTUP. (We feel compelled to mention that *ProSel* consists of a complete set of disk utilities, and not just the program selector.) *ProSel*, for us, made ProDOS friendly. Other extended selectors such as Quality Computers's *EasyDrive* and *RAM-UP, PUPS* ($39.95 from North York Software, 3000 Victoria Park Avenue, Suite 520, North York, Ontario M2J 4Y2, 416-495-0615), and so on have appeared.

Extended selectors based on the shell model are Don Elton's *ECP-8* (included on our June 1989 disk), Dave Lyons's *Davex* ($25 from DAL Systems, PO Box 875, Cupertino, Calif. 95015-0875), and those supplied as part of a programming environment such as *Kyan Pascal*'s *KIX* or *ORCA/M*'s shell. These selectors provide more control over ProDOS than BASIC.System and can also execute files of commands; programmers tend to like them since their functionality can be extended by adding new commands that execute within the framework of the selector.

Symbolic selectors. There have also been graphics-based selectors for ProDOS 8 including *Quark Catalyst* and *MouseDesk*. These use graphic symbols ("icons") to depict files; the files can be selected using a pointing device (ideally, a mouse). These have not survived well; the size and system requirements required to operate these selectors apparently was too extreme for most Apple II users. Apple apparently believed so strongly in this type of interface that they started building computer systems based entirely on this metaphor (the Lisa, the Macintosh, and the IIgs in native mode). Many Apple II users apparently disbelieved so strongly that they elected not to switch to Apple's newer lines of machines.

Our experience with the *Catalyst* and *MouseDesk* programs demonstrated to us that you can either use most of the system resources to produce pretty pictures, or you can use them to do work, and we generally chose the latter option.

The acceptance of the IIgs Finder is wider because, like the Mac, the IIgs was designed with the icon-based interface in mind. With System Software 5.0, the performance of Finder is not so poor as to make us disregard it. There are ways to customize the selection mechanism in the IIgs Finder that makes it attractive to many users; we plan to look at some of Finder's features as a selector next month, as well as mention some IIgs alternatives.—DJD

Miscellanea

GEnie is emerging from the bottle in Europe at last. Bill Louden, general manager of the GEnie service, announced the availability of GEnie in Germany, Austria, and Switzerland on March 23, 1990. The United Kingdom is expected to follow within three months, and then possibly the Netherlands (GEnie also hopes to eventually provide the service in East Germany). ***A2-Central*** will be mailing out sign-up instructions to subscribers as GEnie opens for business in the individual countries.

The local access rate within the first three countries will be $18 per hour for non-prime time and $27 per hour for prime time (GEnie is trying to create a single end-user price structure for all of Europe). Currently, the price is in U.S. dollars.

This announcement follows on the heels of the announcement of Compuserve Forum, a European arm of Compuserve, which began operating out of Berne, Switzerland and Bristol, England on February 1.

Barney Stone has decided to distribute *DB Master Version Five* as shareware. This version of *DB Master* supports up to 200 fields per record, 250 characters per field, and files up to 10

megabytes in size (either using a hard disk or multiple floppy disks). The shareware price for the product will be $45, which entitles the owner to the latest revision of the program, a full instructional manual, and a special offer on *DB Master Professional* (which remains a normal retail product). Barney expects many users to upgrade to *DB Master Professional* once they've seen what *Version Five* will do.

DB Master Version Five requires an enhanced 128K Apple IIe, IIc, IIgs, or Laser 128 computer with at least two 5.25 drives or one 3.5 drive. The program can be installed on a ProDOS compatible RAM disk or hard disk with at least one-half megabyte of available storage. More information is available from Stone Edge Technologies, Inc., P.O. Box 3200, Maple Glen, Penn. 19002, 215-641-1825.

The Spring 1990 *APDALog* (from APDA, Apple Computer, Inc. 20525 Mariani Avenue, M/S 33-G, Cupertino, Calif. 95014, (408) 562-3910 mentions that membership in APDA will no longer be required in order to purchase final copies of development tools and references. The materials will be available through a new Developer Tools Express service, though the service does not include other APDA privileges such as receipt of the quarterly *APDALog* issues and the ability to order prerelease (beta) versions of products.

***Publish-It! 3.0* has several new features.** Although the program is still designed to run on the 128K Apple IIe and IIc models (including the IIgs in "IIe mode"), several enhancements have been added, including the ability to import IIgs super high-resolution graphics. Support for Epson LQ series printers and the Apple Scribe has been added, and color printing is supported.

In addition to the ability to use an expanded memory card to hold the program itself for faster operation, *Publish-It!* now has an option to also load all fonts into memory if you have approximately 400K or more of memory available.

The "Save Settings" command has been enhanced to allow saving defaults for all options in effect at the time of the save, including settings for custom guides, page size, viewing display size, display of pictures in the document, rulers, pen and fill choices, the default font selection, and more. This option does not save your document, so you can use it to determine the startup conditions you prefer for Publish-It!.

New features in the Page Layout section include Go to Page (by number), user defined guide settings, new editing functions (cut, copy, paste, move to front and move to back) on grouped objects, and new object attributes (Don't Print, Locked, Don't Runaround, PostScript, and Color).

By turning off Runaround for an object, you can make text "flow around" the object. Using this in combination with the Transparent option (to allow overlaid objects to be printed) you can create special effects such as dropped capital letters (a single large letter leading into a paragraph).

The PostScript attribute indicates *Publish-It! 3.0*'s new ability to use encapsulated *PostScript* (EPS) files. EPS is a standard format for graphics and text objects consisting of a text file in a standardized *PostScript* command format. By importing an EPS file into a text frame scaled to the desired size of the printed object, the EPS data

---

Ask (or tell) Uncle DOS

One of these days I'll beat the deadline by enough lead time to find the subtle errors, like forgetting the address information for products we mentioned last month. The disk labelling program we mentioned in April ("Disk Labels", p. 6.23) is ***Disk Label Pro****, available from Hardsoft Products, P. O. Box 90012, Honolulu, Hi. 96835, for $30 (postpaid; includes 100 labels for 5.25 disks and 20 labels for 3.5 disks) and* ***Electronic Learning*** *(ISSN 0278-3258) mentioned in "Personnel moves rock Apple" (March 1990) is $23.95 per year (8 issues); the address is P. O. Box 2041, Mahopac, Md. 10541.—DJD*

AppleWorks international

Is there a patch for the alphabetic sort in the AppleWorks 3.0 database such that foreign (e.g., German) characters get sorted correctly? This would be the characters Ä (91), ä (123), Ö (92), ö (124), Ü (93), ü (125) and ß (126). They should be sorted with A, O, U, and S respectively. I'm sure all those who are using other languages would appreciate this.

Does anyone know where I can get a German character generator chip with mouse characters for my II enhancement? The one that comes with the enhancement kit fits American motherboards and has 24 pins; the German character ROM (part number 342-0275 for the upgrade) has 28 pins. Apple Computer in Cupertino informed me by phone that they could not support it; a letter to them brought no reply.

Terrell Smith
Madison, Wisc.

The AppleWorks 3.0 enhancements are really worthwhile, my only grief is that Claris seemingly did not heed your advice to read the internationally different ASCII characters from a table. The $7C character of vertical line shows up as a ô on a IIe using the German keyboard switch. Using the information in ***A2-Central*** on AppleWorks 2.0 I had all $7C characters converted to exclamation marks and thought it easy to do the same to version 3.0. But the disk storage of 3.0 is so different that a huge amount of work is involved. Before I go ahead I would like to know if some reader or organization has already come up with a scheme of where to look and what to change because I hate to reinvent the wheel.

Hans Wolter
Dreieich, West Germany

Please h-h-help. I have been with you for some time and feel you are probably the best source of honest information for my beloved Apple.

I have recently upgraded to AppleWorks 3.0 along with all of my favorite applications, including the TimeOut series.

Yet with all this power on my IIgs and 64 meg hard disk, I cannot yet create a custom dictionary to my liking. What I require is the ability for the AppleWorks dictionary to accept the {, |, }, ..., @ characters to permit me to build a French-Canadian custom dictionary with their equivalent characters é, è, ù, ç, and à. As it now stands, when I invoke the open-apple-V command, AppleWorks accepts most words but once a foreign character is found AppleWorks breaks the word in two. As an example, "Français" would be construed as "Fran" plus "ais".

If the AppleWorks dictionary could recognize these five lower ASCII characters it would permit the construction of a French dictionary that would properly respond to approximately 95% of all spelling errors. The remaining words, those requiring the use of the ^ accent (circumflex) such as "être" and the ¨ (tréma) such as "Noël" would be quickly verified using the old manual system.

I hope you or one of your readers could help me find a patch that would allow a more versatile AppleWorks 3.0 dictionary.

Jean-Guy Mariage
Shannon, Quebec

Is there a way to print a £ on an Apple IIgs through a software package or can it be done through hardware itself without too much difficulty?

Fred Reinstein
San Diego, Calif.

We don't have cures for customizing AppleWorks 3.0 for international characters, but perhaps one of our Gentle Readers may have developed a patch. (We'd like to point all AppleWorks patch hackers to a series by John ***"SuperPatch"*** *Link in* ***The AppleWorks Forum****, which provides details for several AppleWorks 3.0 patches.) Also, you may consider asking one of the companies that has modified AppleWorks for international character sets such as Davka Corporation (see "Foreign Accents, cont.", Jan. 1990, p. 4.95) to see what options they can offer.*

We don't see a long-term solution for text-based programs because their character set is determined by the character ROM in use and the monitor routines which support it, which is part of the computer's firmware design. In the IIe and IIc, models were available with ROM and keyboard support "localized" for specific countries; this was extended to the IIgs by placing several options in ROM and allowing the user to localize the system by selecting the

will be converted into the object on printout and placed in the area defined by the text box. To do this, you enable the *Publish-It! 3.0* PostScript attribute for the text object containing the EPS file while printing to a *PostScript* printer.

You can also print the *PostScript* output for a document to a file on disk, and TimeWorks includes a short section at the end of the 3.0 update manual on how to print these files from the Macintosh systems often found at printing services. Our LaserWriter NT had trouble digesting some of our test files, however.

The program can also now import a IIgs graphic (256K of memory is required on a IIe or IIc) from such programs as *Deluxe Paint II, 8/16 Paint,* and *PaintWorks Gold,* with the limitation that only the first 256 lines of a "tall" graphic will be imported. SHR graphics are imported in color and will be printed correctly in color, though the colors will be distorted on the screen display. Color can also be added to objects and text.

Publish-It! 3.0 now correctly supports the importation of AppleWorks 3.0 word processor files. It also allows the direct use of IIgs fonts, although TimeWorks cautions that the quality may not equal that of the *Publish-It! 3.0* fonts.

We've had our hands on Vitesse's new *Quickie* scanner for the IIgs; Vitesse also now reports they have software to support the scanner on a IIe or II Plus. The *Quickie* consists of a hand-held scanning mechanism connected to the Apple II through an interface card that can be installed in an internal slot.

The IIgs software includes a stand-alone application and an NDA that can access the scanner to import graphic data to the screen while in the IIgs's 640 by 200 super high-resolution screen mode. The software includes selections to allow determining the length of the scanner's path (which can be many feet, though the width of the scanned image is limited to the scanner's dimension of about 4 inches) and options affecting the resolution and representation of the scanned image (black and white, gray scale, dithering) on the display.

The scanner handles gray scale levels (no color), but currently is limited by the storage format of the acquired data, which must be saved as a 640 mode graphic image. The scanner resolution is 100, 200, 300, or 400 dots per inch (selected by a switch on the scanner head).

We found the scanner to be quick and easy to use, and it produced excellent results within the limitations of the 640-mode screen. We'd like to see either added support for using the 320-mode screen (which is capable of 16 gray levels per screen pixel), or a way of saving raw data (as the *ColorEyes* video digitizer does) in such a manner as to allow further "massaging" by other programs. The *Quickie* is $299 from Vitesse, Inc., 13909/2A Amar Road, La Puente, Calif. 91746, 818-813-1270

***HyperLearning Forum* is a new newsletter** available from HyperLearning Network, Box 103, Blawenburg, N.J. 08504, 609-466-3196. Although the first issue (March 1990) expresses a mission of promoting the use of hypermedia products for instruction without naming a specific product, the initial articles seem to be solidly formed around HyperStudio. An individual membership in the HyperLearning Network is $29 for one year, $55 for two years; institutional memberships and other products are also available.—DJD

---

appropriate "Display Language" and "Keyboard Layout" settings in the control panel (the physical layout of the key caps on the keyboard would have to also be rearranged). But text-based programs used on the IIgs still need software that is designed to support the special characters for the language in use.

*Graphics based programs don't have to feel as limited because the character display is not determined by a ROM character set. For example, **AppleWorks GS** can handle the international characters and other characters from the "extended" ASCII character set in its spelling checker.*

*A decision also has to be made where to define these extended characters in the set of ASCII codes; for the IIgs fonts we've looked at, the USA character set occupies ASCII values from 0 to 127, and the international characters are represented as part of the set of ASCII values from 128 to 255. We tested the **AppleWorks GS** database and alphabetic sorting is handled strictly by numerical ASCII value. Most of the "international" characters are accessed by using the prefix keys described in "More keyboard options" (Feb. 1990, p. 6.7a), or by using the option key in conjunction with a character key.*

We became curious about the total range of characters supported, and decided to generate a table for a representative font (Courier). You can use the following Applesoft program to generate such a test file:

```
1000 REM - generate full ASCII table -
1010 LOMEM: 16384: REM brute force
1020 D$ = CHR$ (4)
1030 HEX$ = "0123456789ABCDEF"
1040 TL = 0: REM table length
1100 REM - generate table -
1110 FOR I = 0 TO 255
1120 : GOSUB 2000: REM format current number
1130 : GOSUB 3000: REM create string for character
1140 : GOSUB 4000: REM poke string into memory
1145 :PRINT S$," TL = ";TL
1150 : NEXT I
1155 REM - save table -
1160 PRINT D$;"CREATE ASCII.TABLE,TTXT"
1170 PRINT D$;"BSAVE ASCII.TABLE,TTXT,A$2000,L";TL
1198 END
2000 REM - format I to text string -
2010 I$ = STR$ (I): IF I < 100 THEN I$ = "0" +
I$:
    IF I < 10 THEN I$ = "0" + I$
2020 H$ = "$" + MID$ (HEX$, INT (I / 16) + 1,1)
    + MID$ (HEX$,I - (16 * INT (I / 16)) +
1,1):
    REM do hex string
2030 RETURN
3000 REM - create string -
3010 S$ = I$ + " (" + H$ + ") - "
3998 RETURN
4000 REM - poke string into memory -
4010 FOR K = 1 TO LEN (S$)
4020 : POKE 8192 + TL, ASC ( MID$ (S$,K,1))
4030 :TL = TL + 1
4040 : NEXT K
4050 POKE 8192 + TL,I: REM value
4055 TL = TL + 1
4060 POKE 8192 + TL,13: REM carriage return
4065 TL = TL + 1: REM value
4998 RETURN
```

(The file is written using the BSAVE command because BASIC.SYSTEM clears the high order bit on characters written to a standard TXT file.) The file can then be imported into a program and edited to generate a printout. While editing, watch out for interpretations of certain control codes; for example, the imbedded car-

ASCII value/char	Option	ASCII value/char	Option	ASCII value/char	Option
128 ($80) - Ä	u A	158 ($9E) - û	i u	188 ($BC) - º	0
129 ($81) - Å	A	159 ($9F) - ü	u u	189 ($BD) - Ω	z
130 ($82) - Ç	C	160 ($A0) - †	t	190 ($BE) -	'
131 ($83) - É	e E	161 ($A1) - °	*	191 ($BF) - ø	o
132 ($84) - Ñ	n N	162 ($A2) - ¢	4	192 ($C0) - ¿	?
133 ($85) - Ö	u O	163 ($A3) - £	3	193 ($C1) - ¡	1
134 ($86) - Ü	u U	164 ($A4) - §	6	194 ($C2) - ¬	l
135 ($87) - á	e a	165 ($A5) - •	8	195 ($C3) - √	v
136 ($88) - à	` a	166 ($A6) - ¶	7	196 ($C4) - ƒ	f
137 ($89) - â	i a	167 ($A7) - ß	s	197 ($C5) - ≈	x
138 ($8A) - ä	u a	168 ($A8) - ®	r	198 ($C6) - Δ	j
139 ($8B) - ã	~ a	169 ($A9) - ©	g	199 ($C7) - «	\
140 ($8C) - å	a	170 ($AA) - ™	2	200 ($C8) - »	|
141 ($8D) - ç	c	171 ($AB) - ´	e spc	201 ($C9) - …	;
142 ($8E) - é	e e	172 ($AC) - ¨	u spc	203 ($CB) - À	` A
143 ($8F) - è	` e	173 ($AD) - ≠	=	204 ($CC) - Ã	n A
144 ($90) - ê	i e	174 ($AE) -	"	205 ($CD) - Õ	n O
145 ($91) - ë	u e	175 ($AF) - Ø	O	206 ($CE) -	Q
146 ($92) - í	e i	176 ($B0) - ∞	5	207 ($CF) -	q
147 ($93) - ì	` i	177 ($B1) - ±	+	208 ($D0) - –	-
148 ($94) - î	i i	178 ($B2) - ≤	,	209 ($D1) - —	
149 ($95) - ï	u i	179 ($B3) - ≥	.	210 ($D2) - "	[
150 ($96) - ñ	n n	180 ($B4) - ¥	y	211 ($D3) - "	{
151 ($97) - ó	e o	181 ($B5) - µ	m	212 ($D4) - '	]
152 ($98) - ò	` o	182 ($B6) - ∂	d	213 ($D5) - '	}
153 ($99) - ô	i i	183 ($B7) - ∑	w	214 ($D6) - ÷	/
154 ($9A) - ö	u o	184 ($B8) - ∏	p	215 ($D7) - ◊	V
155 ($9B) - õ	n o	185 ($B9) - π	p	216 ($D8) - ÿ	u y
156 ($9C) - ú	e u	186 ($BA) - ∫	b		
157 ($9D) - ù	` u	187 ($BB) - ª	9		

Extended ASCII Key Equivalents

riage return (ASCII 13) will generate a blank line in your chart printout unless you edit that character out.

We edited our version of the table to encompass the printable characters accessed either by using an "option-character" (both keys at once) command, or by a combination of an option-character command followed by a second character. For example, ¬ is generated by option-l, ¡ by option-1, and ¨ by option-u followed by a (normal, non-option) space.

There are a few idiosyncrasies. First, The characters represented in our table are for the Courier character set; other character sets may differ in the content of completeness of their character set representations. For example, Cairo is composed of graphic characters rather than the normal printable characters we interpret as readable text.

*Second, a font editor will generally show you the complete character set for a font, but it may not correlate directly to what you actually have available in your program. For example, **AppleWorks GS** seems to disregard all control character codes as printable characters; control-M (ASCII 13) is executed as a carriage return as expected, but ASCII characters $11 (control-q), $12 (control-r), $13 (control-t) and $14 (control-s) from the Chicago font were not displayed with **AppleWorks GS** in the printable forms indicated by Beagle Bros' **GS Font Editor**:*

⌘, ✓, , ◆ (in the Chicago font)

(To add to the confusion, the Macintosh has elected to display ASCII $11 as the Mac option key "cloverleaf" instead of the open-apple character shown for the same character in the IIgs Chicago Font we looked at.)

*Third, the font you have on screen may not correlate to the font you see on the printout. Printing the document using the IIgs ImageWriter driver (which sends a bitmap of the document image to the printer) gives a hardcopy that matches the on-screen appearance. Printing the same document to the LaserWriter results in several character substitutions; as we've learned to expect with Apple's desktop environment, what you see is not always what you get. For example, the ≠ (option-=) character visible on screen did not print on the LaserWriter from the IIgs. The reason the substitution occurs is that the LaserWriter driver is fed the QuickDraw II (rather than the bitmap) form of the document which includes separate definitions for the graphics and text elements of the page image as displayed on the screen. When the text is processed into **PostScript** it is converted to ASCII character codes that the LaserWriter is expected to print as the characters we see displayed in the text on screen. (We've seen dissimilar weirdness on the Mac in different circumstances. We mention this to indicate that non-WYSIWIG output is not a problem unique to the IIgs.) Unfortunately, the LaserWriter character codes apparently don't match up. The ImageWriter image is "correct" since it is essentially a graphic printout of the composite image displayed.*

A table like our "Extended ASCII Key Equivalents" on the previous page has a practical purpose: it gives you a reference to know which characters are available on screen and in your printout, and (for the selected codes we've included) how to generate them from the keyboard. For the IIgs, you need to use the option-key mapping, which does not correlate to the ASCII order of the keys in the ASCII chart. So we have an additional column in the table that shows the option-character equivalents for the (US) keyboard layout, along with the few that also have a shifted equivalent (option-shift-character). We can't reproduce the table for all foreign keyboards, but if you can locate a Macintosh user manual you may find it contains a keyboard diagram with similar information (our Macintosh SE manual had the keyboard definitions on page 139). Looking at the table, it turns out that the £ character can be generated by the option-3 keyboard sequence, and the German essett (ß) character by option-s (notice that is a lower case "s").

*The extended characters can also be generated with TimeOut **SuperFonts**; check your manual for information regarding the <x1>, <x2>, and <x3> commands for accessing ASCII ranges of 32-128, 128-191, and 192-255 respectively. The offset is from the ASCII equivalent in the range of @ (ASCII 64) to " (ASCII 254). We see from our ASCII chart that £ is ASCII 163 which lies in the <x2> range; 163 minus 128 is 35. Adding 35 to 64 gives us "c" (lower case) as our normal ASCII equivalent, so we use "<x2>c<x1>" as our **SuperFonts** sequence to generate £ on the printout. The exception is ASCII values 191 (our ø) and 255 (undefined for our chart); normal ASCII character 127 is the non-printable "Delete" key, so "?" is used in it's place.*

After hours investigating this, we aren't sure we have all the loose ends tied down; for example, we tested this all on our decidedly USA IIgs. We have included the tables so that others can experiment; we haven't seen the information compiled in one place anywhere else and we're among the people who need to use it occasionally. Now at least we'll be able to find it!–DJD

AppleTalk and *WordPerfect*

I like *WordPerfect* and I need a word processor like it because of its footnote and hyphenation possibilities. And with the latest version everything worked fine, until I tried to print with an Apple LaserWriter II NT.

First, I printed via the AppleTalk network. It printed, but the results were bad. The worst printing happened with accents.

Then I tried to print via the IWEM (ImageWriter Emulator) to slot 7; nothing happened.

I hope someone can help me with this. I also have written to WordPerfect Corporation, but it could be that you or a reader might have the right answer. I have 1800 pages in *WordPerfect* format with (French) accents and therefore I'm longing for a solution.

I would also like to know if it is possible to make the LaserWriter print in the emulation mode with foreign characters (like the way it can be done with the real ImageWriter).

Jürgen Wöretschofer
Maastricht, Netherlands

*We do not use **WordPerfect** at all at **A2-Central**; possibly a reader has a suggestion.*

The LaserWriter's ImageWriter emulation does support foreign characters;. Since you can't open the lid and change the DIP switch settings as on the ImageWriter, you need to send the appropriate software commands as part of your printer initialization. For example, the sequence to use the Danish characters is Escape Z Control-E Control-@ Escape D Control-B Control-@.—DJD

Math coprocessors

Please consider reviewing math coprocessors for the IIe and IIgs. There's a couple on the market: *FPE* (*Floating Point Engine*) and the Applied Engineering *FastMath*.

My interest is IIgs specific. Will a coprocessor speed up screen display and computation of objects in object drawing applications such as *Top Draw*? Does it improve performance of *AppleWorks GS* spreadsheet, graphics, and database modules?

Dan Rencher
Cincinatti, Ohio

*For an idea of the relative performance of the hardware for the **FPE** and the **Fastmath**, check the times given in the January 1990 "Miscellanea" column using the **Beagle Compiler's** interface routines. The **FPE** was the clear winner, but those not interested in the utmost in performance may wish to factor the lower price of the **Fastmath** into their purchase equation.*

*As your interest is IIgs-specific, the **Floating Point Engine** is the only realistic alternative. **FastMath's** hardware does not provide the SANE (Standard Apple Numerics Environment) compatibility necessary to augment the IIgs' or (classic) AppleWorks' use of the SANE model for floating point calculations.*

*Be clear on the point that a floating point coprocessor only speeds up calculations. The **FPE** will accelerate **AppleWorks GS**'s spreadsheet recalculations, for example, because the **FPE** includes a routine to cause calls to the IIgs SANE routines (by any program) to be routed through the **FPE** instead. Graphics that depend on a great many SANE calculations (such as a fractal program that calls SANE to calculate the points to plot) will be sped up by the virtue of quicker calculations, not by any acceleration in the actual plotting of points to display the graph. Most general painting or drawing programs won't use SANE to calculate the form of the object they are commanded to draw because the object's image can be rendered more quickly and adequately for display purposes by using other techniques.—DJD*

Carry me away

Do you know of any company that makes carrying cases for the Apple II like they have for the Mac?

Paul Christianson
Lancaster, Calif.

We haven't seen a specific IIgs case advertised. The IIgs monitor and CPU might fit in cases designed for a compact Mac IIcx style of system (the IIgs components are a bit smaller than their Mac counterparts).

*Targus manufactures a soft case called the **Lappac 2 Deluxe** ($99.95 suggested retail for woven vinyl, $219 for the leather version) that has interior dimensions of 17" by 14.6" by 4" intended for a wide range of laptop and portable systems (including the IIc models). Such a case wouldn't house the IIgs monitor but would probably be adequate to carry the CPU, an external drive or two, and the necessary cables. The case has a padded internal divider that splits the large compartment into two sections; one could be used for the CPU and the other for a 3.5 drive and cables. The interior of the "lid" has pockets specifically*

designed for 3.5 disks, business cards, and writing implements, as well as a larger pocket for files. There are outer pockets designed to hold large items, possibly a modem or external drive. Targus also sells other cases; they sell through dealers, so you may find the ***Lappac 2*** *case at dealers that carry MS-DOS computers and accessories. Mac Connection, 14 Mill Street, Marlow, N.H. 03456, 800-334-4444 or 603-446-7791) sells Targus cases for the Mac IIcx ($75) and the ImageWriter II ($45); the IIcx case might be suitable for the smaller IIgs CPU and keyboard.*

Another source to check might be your local music instrument store; the places that sell keyboards, guitars, electronics, etc. to professional musicians. Those instruments need hard shell cases to protect them "on the road", and some of the case suppliers will actually configure a case to match your requirements. Audio-visual equipment dealers may also be a place to check.—DJD

Disk identification

Back in Volume 2, Number 12 (January 1987) you showed us how to execute Smartport commands. This information permits me to detect the presence of a RAM disk that uses SmartPort protocol. In Volume 3, Number 8, Alan Silver shows how to detect a "RAM disk" that does not use the SmartPort protocol. Both of these ideas work fine.

My problem is this: once I have detected a SmartPort RAM volume (under ProDOS 8) how do I find its ProDOS volume name so that I can do a GET_FILE_INFO MLI call to find out how many free blocks the volume has? (The SmartPort device status call only returns the total number of blocks, not the number of blocks used or the number of blocks free.) I need to know if the RAM volume has sufficient unused storage space for my program's temporary files.

The method I am currently using is to do a GET_FILE_INFO call for each online volume and compare the "total blocks" of that volume with the number of blocks that was returned in the SmartPort Device Information Block for the SmartPort RAM disk. When they're equal, I assume I've found the SmartPort RAM volume. This technique dares to assume that the RAM volume will never be exactly the same size as any other online volume. So far I've lucked out, but sooner or later one of my programs is going to run up against a RAM volume that is the same size as an online floppy disk, or other block-oriented device, and my method will byte the dust. What can I do?

Robert C. Moore
Laurel, Md.

Apple has released two ProDOS 8 Technical Notes you should read: #20 ("Mirrored Devices and SmartPort") and #21 ("Identifying ProDOS Devices"). (Apple Technical Notes are wonderful things and are highly recommended to serious programmers.) These clarify the identification of ProDOS (disk) devices.

Basically, if you have isolated where the RAM disk is so that you know the slot and drive assigned to it, then you have it's ProDOS unit number. The unit number format is DSSS0000 (in binary), where D is 0 for drive 1 or 1 for drive 2, and SSS is the three bytes representing the slot number. Therefore, if you have a RAM disk that appears in slot 4 as drive 1, the unit number would be 01000000 ($40 in hex; 64 decimal).

The MLI ONLINE ($C5) call requires the unit number as part of its parameter block. If you use the specific unit number for the device of interest, ONLINE will return the volume name for the device in the buffer you specify. As an example, executing:

```
        JSR  MLI       ;execute MLI command
        DFB  ONLINE    ; online call
        DA   OLPARMS   ; our parameters
        BCS  ERROR     ;handle any error
        RTS            ;return
OLPARMS DFB  $02       ;two parameters
        DFB  $40       ;slot 4, drive 1
        DA   OURBUFR   ;our buffer location
```

would return the volume name information at location OURBUFR. The actual form of the information is a byte at OURBUFR which contains the unit number of the device in the high-order four bits, the length of the name of any ProDOS volume present in the device in the low-order four bits (if the device is empty, this value is zero), and the ASCII text (high bit clear) for the volume name beginning at the byte following OURBUFR. (In this case, the text of the volume name does not contain the leading "/" normally associated with ProDOS volume names.) If things don't go well, Possible errors are $28 (no device connected), $52 (not a ProDOS disk), and $27 (disk I/O error).

With the volume name, you can now do a GET_FILE_INFO call (See "Butterflies turn to worms", Oct. 1985) to return the information for the volume directory "file". For a volume directory, the field that normally would indicate an auxiliary type actually indicates the total number of blocks on the volume. Subtracting the blocks used from the total blocks gives, of course, the blocks free.—DJD

IIgs programming in C

How about carrying some books on programming in C? Especially with reference to the IIgs toolbox.

Glenn Goldstein
Drexel Hill, Pa.

We know of a few good books on C programming in general; we see most of them on the shelves at normal bookstores so we haven't bothered to try and sell them ourselves. We've stuck primarily to Apple II specific books that might be somewhat harder to locate locally.

*There is no single "IIgs Toolbox programming in C" reference that is topically current that we are aware of. Overall, you probably shouldn't assume that such a book will exist in the near future; books take several months to write and at the rate the IIgs System Software is changing right now any book written from current knowledge would probably not be comprehensive by the time it is finished. The IIgs C compilers available now (**APW C** and **ORCA/C**) are also still being improved.*

This does not mean there is a lack of hope; books on toolbox programming do exist (see "Desktop programming help"; p. 6.21 of last month's issue); though most examples are in assembly language, by and large the principles apply to high level languages as well. Tool calls do not look that distinct between various languages, in every case it is a matter of allocating the necessary data structures and just calling the correct routine.

The important factor is understanding the language you are using (be it assembly, BASIC, Pascal, C, or other) well enough to be able to create the program logic necessary to provide the framework and environment for making the toolbox calls. This means you should learn the language itself first before trying to perform functions specific to a particular system. It's the same principle as learning chemistry before trying to make nitroglycerin.

*Apple's **Programmer's Introduction to the Apple IIgs** contains the complete source code for the demonstration **HodgePodge** application in assembly language, Pascal, and C. Pascal is used for the text of the book, but you could flip back to the index and look at the equivalent C code. Cecil Fretwell has converted the code for **Programming the Apple IIgs in Assembly Language** to C source on disk, and you could also get that disk, print out the source files, and follow along with the book with C source in hand. The crucial step is to know enough C to be able to understand the examples.—DJD*

IIgs to II BASICally

I'm writing a role playing game of my own in *TML BASIC*. I know this works with the Apple IIgs, but can you recommend any language or program like TML that works on earlier Apples, i.e. IIc, IIe? I'd like my game to be available for these people, too.

Harold Reynolds
Minnetonka, MN

*If you're looking for a BASIC compiler, the 8-bit Apple II options are **Micol Advanced BASIC** for the Apple IIe and IIc, **ZBASIC**, or using Applesoft and the **Beagle Compiler**.*

*Actually, since you have a IIgs, there is another alternative: Morgan Davis has announced **MD-BASIC**, a "preprocessor" which allows using the **ORCA/M** or **APW** IIgs assembly environment to translate a structured BASIC source file into Applesoft, which can then be compiled with the **Beagle Compiler**. As an example, the program:*

```
'== Trivial sample for MD-BASIC ==
'constant definition
#define KeyDataReg      -16384
#define KeyStrobe       -16368

'macro definition
#define WaitForKey      WAIT KeyDataReg,128
#define ClearKey        POKE KeyStrobe,0

Main:
   TEXT
   HOME
   ONERR GOTO FixError
   REPEAT
      PRINT "Key?: ";
      WaitForKey
      Key = PEEK(KeyDataReg)
      ClearKey
      IF Key < $A0 THEN
         PRINT "  ASCII code = ";Key
      ELSE
         PRINT "  Character = ";CHR$(Key)
      ENDIF
   UNTIL Key = $8D          'wait for <Return>
   PRINT "That's all!"
   POKE 216,0               'cancel ONERR
   END                      'Main'
FixError:                   'trap control-C
   RESUME                   'just return
```

translates to this:

```
100 TEXT
110 HOME
120 ONERR  GOTO 250
130 PRINT "Key?: ";
140 WAIT  - 16384,128
150 KEY =  PEEK ( - 16384)
160 POKE  - 16368,0
170 ON  NOT (KEY < 160) GOTO 200
180 PRINT "  ASCII code = ";KEY
190 GOTO 210
200 PRINT "  Character = "; CHR$ (KEY)
210 IF  NOT (KEY = 141) THEN 130
220 PRINT "That's all!"
230 POKE 216,0
240 END
250 RESUME
```

*after a pass through the **MD-BASIC** preprocessor with optimization "off". There's even a "decompiler" to convert existing Applesoft programs to **MD-BASIC** source code, and several advanced features to eliminate some of the pitfalls of normal Applesoft (indentation, no line numbers, extended variable names, preprocessor macros, and so on) as can be seen in the sample above. The **MD-BASIC** preprocessor is a shell utility that works on files edited with the **ORCA/M** IIgs or **APW** editors, hence the need for the **ORCA/M** or **APW** environment. **MD-BASIC** also includes **AmperWorks**, a set of ampersand ("&") enhancements to Applesoft. **MD-BASIC** is $49.95 from the Morgan Davis Group, 10079 Nuerto Lane, Rancho San Diego, Calif. 92078-1736, 619-670-0563.*

A2-Central™

[redacted]

Edited by:

Dennis Doms

with help from:

Tom Weishaar	Sally Dwyer	Dean Esmay
Joyce Hammond	Jeff Neuer	Jay Jennings
Tom Vanderpool	Jean Weishaar	

A2-Central—lifted ***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): $28 for 1 year; $54 for 2 years; $78 for 3 years. All back issues are currently available for $2 each; bound, indexed editions of our first four 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 $84 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.

A2-Central is sold in an unprotected format for your convenience. You are encouraged to make back-up archival copies or easy-to-read enlarged copies for your own use without charge. You may also copy ***A2-Central*** for distribution to others. The distribution fee is 15 cents per page per copy distributed.

WARRANTY AND LIMITATION OF LIABILITY. I warrant that most of the information in ***A2-Central*** is useful and correct, although drivel and mistakes are included from time to time, usually unintentionally. Unsatisfied subscribers may cancel their subscription at any time and receive a full refund of their last subscription payment. The unfilled portion of any paid subscription will be refunded even to satisfied subscribers upon request. MY LIABILITY FOR ERRORS AND OMISSIONS IS LIMITED TO THIS PUBLICATION'S PURCHASE PRICE. In no case shall I or my contributors be liable for any incidental or consequential damages, nor for ANY damages in excess of the fees paid by a subscriber.

ISSN 0885-4017

Printed in the U.S.A.

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

*The suitability of any of these alternatives depends closely on the number of specific **TML BASIC** features you may have used regarding access of the IIgs toolbox, and for specific uses such as graphics. For example, **Micol Advanced BASIC** and **ZBASIC** both have some graphics commands extensions to draw circles and other objects on the graphics screen; in Applesoft (and **MD-BASIC**), such features are absent and you would have to write your own subroutines. (**MD-BASIC** has an "#include" instruction that allows subroutine source to be imported into a file at "compile" time.)—DJD*

IIgs modem help

I would like to add a modem to my IIgs to take advantage of your GEnie service. I have *AppleWorks GS* which has a communications module. Please offer suggestions or literature which would guide me in firmware, hardware, software, etc. and especially installation procedures.

Peter Orio
Tweksbury, MA

Our back issues have some information on modems in general and connecting to GEnie in particular; the pertinent articles are "Mainframe GEnie at your service" (Feb. 1988), "Crossing telephones with computers" (Mar. 1988), and "Twisting talk into data" (May 1988).

*With your modem installed, the **AppleWorks GS** manual should be adequate to get you up and running, though if you're looking for practical tips you might find another reference handy. We've heard good things about Que books, and a look at their new **Using AppleWorks GS** convinced us to include it in our catalog.*

*One note you won't find in the books; System Software 5.0.2 has a small bug in the SCSI driver that can cause **AppleWorks GS** (or any native IIgs communications program) to die with a error indicating the program has received too many interrupts. Claris has some suggestions on dealing with this problem for now, which we anticipate is on the "redress" list for any future system software release. The SCSI drivers for Apple's new High-Speed SCSI Card (which also includes a minor GS/OS revision to version 3.2) apparently still allow the error to be generated, but the system will allow you to recover rather than forcing a reboot.—DJD*

Don't do (too many) windows

I own a IIgs and am always looking out for new graphics and font programs. Recently I incorporated several new fonts from *GraphicWriter* into both *AppleWorks GS* 1.0 and 1.1. In both cases I have run into the same problem: I can't print (I either get "Can't print this document" or "Not enough memory to complete", etc.) if I have more than one window open. I was using a graphics page but it only had minimal black and white art and a name written five or six times using different fonts. Is this problem due to these fonts being in graphic mode? Any suggestions?

John Reed
Simi Valley, Calif.

*We mentioned the problems of memory constraints last month ("Serious errors", pp. 6.11-12). One thing we didn't mention is that open windows are one of the things that can eat memory, in addition to multiple fonts. If memory is very tight and you have an idle (or expendable) document window open, close it; it may get you through the next operation, but only if the difference is a few thousand bytes (windows don't use **that** much memory).*

*Fonts, graphic objects (including those used by the system, such as windows), and many other items can affect memory use. **AppleWorks GS** allows you to see how much memory you have free as part of a status report shown when you hold down the Option key and pull down the "About AppleWorks GS..." item from the Apple menu. If you are chronically short of memory (after stripping out anything expendable in the way of desk accessories, RAM disk space, etc.), the only permanent answer is to get a larger memory expansion card.—DJD*

Too much power?

Well, I had a quiet chuckle when I read Vern Mastel's article in last month's newsletter about relative speeds and capabilities of the MS-DOS and Apple II systems.

I am one of a team of nine people scattered around Victoria engaged in community development work. For the last three years I have used a computer for numerous tasks involving lots of word processing, low level desktop publishing, databases, statistical analysis and plenty more.

Recently, I prepared a paper on the computer needs of my fellow workers with a view to having them equipped for the task. The paper outlined our needs, based on what I had been doing, and was submitted to the computer experts of our organization for appraisal. The "experts" came back with a recommendation that we be equipped with MS-DOS compatible desktop 286/386 with VGA monitors, 110 megabyte hard disks and laser printers. Sounded pretty good to me. The only fly in the ointment, however, was that I had been doing all my work on a 128K IIe, standard monitor, 20 meg SCSI drive (although my software runs happily off floppy disk) and Olympia dot matrix printer.

There's no doubt about it, the potential of the more powerful machines is mind boggling. Unfortunately I can't afford the luxury of a boggled mind, I've got too much work to do in the real world. The new machines suggest a learning curve like the north face of the Matterhorn; okay for masochists but not much fun for the rest of us.

I think we'll probably end up with laptops which makes more sense (we're a pretty mobile bunch), but I am as keen as mustard to get that Hewlett Packard laser printer running under AppleWorks 3.0, *DB Master Pro*, and *Printrix*!

Incidentally, the April newsletter was great—more positive than I've seen in quite a while, plus lots of useful tips.

Ian Wright
Ballarat, Vic.

*It looks like the portable IIc would be a good choice for you, too. Unfortunately, we learned that Roger Coats can no longer obtain the **C-Vue** LCD screen (mentioned in last month's "Permanent portable IIc") in stock, which pretty much kills the use of a IIc as a battery-powered portable.–DJD*