The few, the proud...the life and times of an Apple II programmer

By Jay Jennings

How would you like to have more money than you know what to do with? Do you want friends, family, and total strangers looking at you with respect and admiration? Could you handle first-class travel to exotic world locations? If you've answered "Yes!" to the above questions, you're reading the wrong article. If, however, you'd like to make a little money and have some fun on your Apple II computer, read on.

A lot of you reading this article know how to program the Apple II in one way or another. This one's for you. I'll show you how to become a professional Apple II programmer. I'll also show you how to become a semi-professional Apple II programmer. That's someone who has a "real" job, but makes money writing professional quality software on the side.

"But wait!", I hear you crying. "Professional software is something like AppleWorks or *HyperStudio!* I can't do that!" That's okay, I can't either, and I'm still a professional programmer. (Okay, I could probably write those, but it might take me a few days.) There are markets for little software packages and markets for big software packages. And it's relatively easy to get into either.

If you want to become a computer programmer for NASA, you might need to go to school for a few years. Fortunately for us, the Apple II world is a bit more intelligent. I went straight into programming from a career as a performer, without college, even without high school. What counts in the Apple II world is what you can do (and, as you'll see later, who you know).

For purposes of this article we're going to ignore shareware as a means of making money. It can be an important step in making a name for yourself but there's no money there, so it can't be thought of as a job (at least not in the Apple II world). And real programmers have to think about such mundane subjects as bills, rent, food, etc.

The two types of programming jobs I'll be focusing on in this article are contract programming and in-house programming. When I go to work Monday morning, I walk into a building filled with 15 programmers. We're all in-house programmers. We're salaried employees whose job it is to write computer programs. Yeah, they pay us for playing with computers. Is that great, or what!

Contract programming, on the other hand, is usually done from your home (or home office) for a company that is not your employer. One of my first programming jobs was to rewrite an interface for a conversion program distributed by Roger Wagner Publishing. RWP is in San Diego and I lived in Alaska at the time. I sat in my house by the creek, watched snowflakes fall from the sky, and wrote a program. In return, RWP paid me.

There are pros and cons to both contract and in-house programming. As a salaried employee in a company you usually have more job security, health insurance, etc. Writing software as an inde-pendent contractor means you can keep your own hours, choose the projects you'll work on, etc. Most of the in-house programmers I know actually got their start as contract programmers, so we'll use that as our first goal.

First things first. To program for fun and profit you need a computer. The best platform is a fully-stocked Apple IIgs. The next best platform is whatever you have available. I started programming on an Apple IIc with the internal floppy drive. Professional programming can be done with very little hardware but the more you have the easier your job will be.

The best CPU for Apple II development is an Apple IIgs. This is true even if you're only going to be developing 8-bit software. The tools available to you on the IIgs are far superior than the 8-bit tools you can find. If you have a IIgs you should have at least 2 megabytes of RAM. More RAM is definitely better. If you have an 8-bit Apple II, 128K RAM should be your minimum. I had 1 meg on my IIc and found the resulting RAM disk invaluable for development.

Both 3.5 and 5.25 floppy drives are handy. It's possible to get by with only one or the other, but with both so prevalent in the Apple II world you could find yourself stuck with a floppy and no way to read it at some point.

Can you create software without a hard drive? Yes. Can you create software without a hard drive and keep your sanity? No. (Well, if you're only developing 8-bit software you might be able to keep the men in the white coats at bay.) A hard drive is almost a necessity for Apple IIgs development. How large does the hard drive need to be? A good rule of thumb is that however large it is, it isn't big enough. If
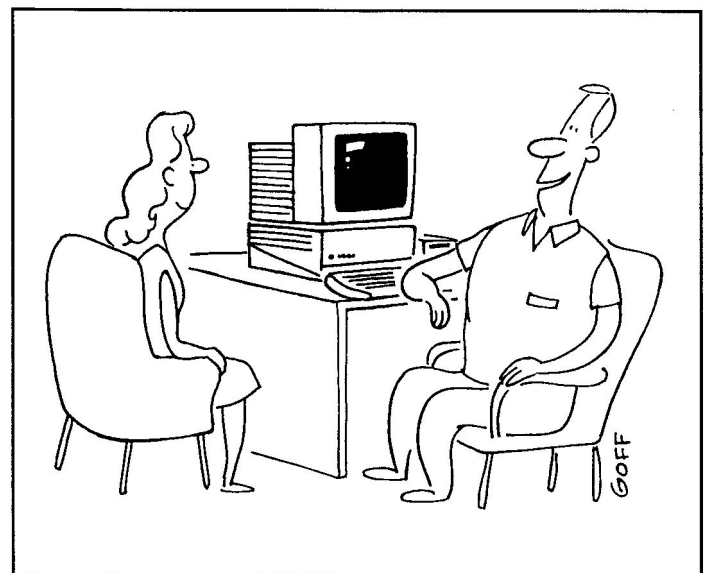
"THIS NEW SCREEN SAVER IS VERY RESPONSIVE. EVERY TIME MY MIND GOES BLANK, SO DOES THE SCREEN."

you need something a little more precise than that, get an 80 or 100 meg hard drive. It'll take longer to fill than a 40 meg drive (which seems to take about 14 minutes to fill).

An accelerator is also a nice addition to your system. I worked without one for years before breaking down and buying one. It's nice, but not a necessity (unless you're using a slow compiler on the IIgs - more about that later).

If you have the extra money get a printer. Being able to look at a program listing on paper while scanning it on the screen is a real time saver. Oh, and buy a monitor. You'll need it unless you're psychic (in which case you can stop reading right now because you already know how this article ends).

And the last piece of hardware that you need for software development? A modem. That should actually go at the top of your list. After the CPU, but before hard drives, printers, etc. Why? One good reason is that you can't make money programming unless you know what programming jobs are available. And the modem is the shortest distance between you and a job. More about modem magic later.

Besides hardware you're going to need software. It's hard to write programs without assembler, compilers and editors. Yes, you can write Applesoft programs without extra software, but you can do it faster and easier by investing in a few tools.

Note! I'm going to be talking about specific pieces of software in this article. If I say they're good it's because I like them. However, there are some tools that I won't be mentioning. Why? Because my mother always told me not to say anything at all if I couldn't say something nice. The opinions expressed in this article are those of the author and not necessarily those of any other person, living, dead or undead.

For Applesoft programming, I highly recommend *MD-BASIC* from The Morgan Davis Group. It's a translator that runs on the Apple IIgs that turns source code into Applesoft code. For more information on *MD-BASIC* see the January issue of *A2-Central*.

If you're going to write Applesoft programs on an 8-bit Apple II you should get something like *Program Writer*, by Alan Bird. *Program Writer* allows you to use a full-screen editor for creating Applesoft programs. It also renumbers lines, among other handy tricks.

You say you're programming on an 8-bit Apple II but want something a little more powerful? There's always my favorite language, assembly. I started out using the *ORCA/M Assembler* from The Byte Works. Besides being a full-featured assembler it includes a very powerful shell environment. In fact, I used to boot straight into the *ORCA* shell on my IIc and used that as my launching program. The *Merlin* assembler from Roger Wagner Publishing is also available (more on that in a few moments).

What about languages in between Applesoft and assembly? In my opinion, there aren't any worth using. A couple years ago I would have recommended *ZBasic* (I actually still use it from time to time myself) but Zedcor, the parent company, dropped out of the Apple II world and *ZBasic* isn't supported at this time. Using a development language that's not supported by the manufacturer isn't really very smart.

In the 16-bit world of the Apple IIgs you have a few more choices. If you're looking for a high-level language your best bet is HyperTalk, the language in *HyperCard IIgs*. Everytime I start thinking about writing a BASIC for the IIgs I realize that most of what I want is already built into HyperTalk. It's slow and it's big, but it's also very powerful.

Moving up the ladder (or down, depending on your perspective) we come to C and Pascal. The Byte Works makes a compiler for both languages. They are full-featured, ANSI standard versions of these popular languages. Using C or Pascal makes sharing code between different platforms much easier than in most other languages. As an example, just recently Bill Heineman, author of such IIgs creations as *Out of This World*, ported a complete IBM PC game to the IIgs in about a week. Most of the C source code ported straight over with no (or very little) tweaking.

If you use C or Pascal you may want to install an accelerator in your IIgs. Otherwise, due to the amount of work compilers must do, you'll find yourself with lots of time for reading, going for walks, or making sandwiches while your program is compiling.

Finally, you can use assembly language. With assembly language your program can be as small and fast as possible. Of course, your development time might be a little longer with assembly so you'll have to decide what's more important for you. The Byte Works makes a very powerful development environment and assembler. ORCA/M is the assembler, *ORCA/C* is the C, and *ORCA/Pascal* is the Pascal. Install the environment once, install the different compilers/assemblers, and you have an unbeatable combination of languages. Well, unless you're the kind of person who wants to program in assembly only. In that case...

My assembler of choice at this point is *Merlin 16+* from Roger Wagner Publishing. To tell you the truth, I used to hate it with a passion. I thought the interface looked like it was designed by a mentally deficient 13-year old, the full-screen editor was too weird for words and it didn't include a shell. So why did I switch? Roger Wagner offered me a contract programming job — but the "catch" was that I had to do it in *Merlin*.

Okay, I have my standards. I know what it means to stand up for the things you believe in. But this was **the famous** Roger Wagner talking to **the nobody** Jay Jennings! How could I turn that down? Since this could be the foot in the door that I was looking for, I bit the bullet, gagged, and used *Merlin*. And found that **blazing speed** was what was missing in my life.

No, speed isn't everything. The *ORCA* environment is more open-ended. The *ORCA* environment allows you to write part of your program in C or Pascal and the other part in assembly. But *Merlin 16+* is so much faster than *ORCA/M* that it's not even fair to compare the two in terms of assembly times. My programming style (make a change, check it out, make a change, check it out, etc.) makes a fast assembler a must. If you want to mix and match languages, *ORCA's* the way to go. Or you can do what I do. Get both development tools and use one or the other depending on the current project.

You'll also need a resource compiler/editor. *Rez, Genesys* and *Foundation* are the only ones available at this time. I use *Genesys* to flesh things out (windows, menus, etc.) and then create *Rez* code out of that. If you can only afford one, get *Rez*. It's much more complicated than the others, but it will also do just about anything you need it to. *Rez* is available as part of the APW Tools and Interfaces package. You can get that from Resource Central. The Byte Works also includes *Rez* in some of the packages they sell. Time out for a special note. The APW Tools and Interfaces is the only thing you should own that includes the letters APW. You don't want APW/C or the APW assembler. They're no longer supported and they're slower than they have a right to be.

Okay, you have the hardware and software that you need. Are you ready to begin making your fortune? Not quite. You need something that will tell you the ins and outs of the Apple II you're going to program.

Can you program the Apple II without owning any reference manuals? Yes. Can you do a good job without them? No. You can spend around a bazillion dollars on Apple II books (and I have) but you don't need to. For $100 (more or less) you can get everything you absolutely need. Anything else is gravy.

Here's a list of books that you'll want to get for programming the 8-bit Apple II:

> *Apple IIe Technical Reference Manual* by Apple
> *Apple IIc Technical Reference Manual* - 2nd Edition by Apple
> *Exploring Apple GS/OS and ProDOS 8* by Gary Little

And if you want to program in Applesoft:

> *Applesoft BASIC Programmer's Reference Manual* by Apple

BASIC Programming with ProDOS by Apple
Books needed for programming the Apple IIgs:
 Apple IIgs Toolbox Reference Volume 1 by Apple
 Apple IIgs Toolbox Reference Volume 2 by Apple
 Apple IIgs Toolbox Reference Volume 3 by Apple
 Programmers Reference for System 6 by Mike Westerfield
 (commonly referred to as Toolbox Reference Volume 4)
 Apple II Tech Notes by Apple
 Apple IIgs GS/OS Reference Manual by Apple
 Apple IIgs Firmware Reference by Apple
 Apple IIgs Hardware Reference by Apple

There are a lot of other books available. If you have the money, get them. The more sources of information you have, the better off you'll be.

If you need to learn assembly language, Roger Wagner Publishing has two books available, *Assembly Lines: The Book*, and *Apple IIgs Machine Language for Beginners*. If you need to learn C or Pascal, The Byte Works has a course available that will teach you the language, and a course that will teach you toolbox programming in that language. If you need to learn Applesoft, the afore-mentioned Applesoft books should be enough to get you on your way.

Earlier I mentioned the need for a modem. The best investment I ever made as a programmer was getting a GEnie account. Every programming job I've ever done has come as a direct result of a contact on GEnie. The shortest path to becoming a "someone" in the Apple II world is via GEnie. Yes, there are other online services available, and they're okay, but the cream of the crop for Apple II people is GEnie (in my humble, but totally accurate opinion).

Besides the networking aspects of an online account, you have access to the best brains in the business. If you're having a problem understanding an algorithm, or stuck with a bug you can't track down, or need advice on assemblers and compilers, you can probably find the answer online. Usually within a few hours.

Okay, you have the hardware, the software, the manuals, and access to all the **big name** people in the Apple II world. Now what? Sit back and wait for the money to start rolling in? Not quite. In order to make money writing software, you have to - write software.

You want to get your name in front of people. You want people to see that you can create software that works. So the first thing to do is come up with a program that you can write in a relatively short time (1-3 weeks). Oh, and coming up with a program that people will want to use is also a good idea. Writing an Ultima VII clone would be cool, but not very practical. You want to get something done sooner, rather than later (if you can do an Ultima VII clone in 1-3 weeks, stop reading - you can find you a job **anywhere**).

After you write one program you need to test it to make sure it works (you might be surprised at how many programmers skip that step--on the other hand, you might not be) and then upload it to as many online services as you have access to. Include documentation with the program so people will not only know what it's supposed to do, but how to do it. Your name should be in the docs and, of course, on the front screen of the program.

Do you need to copyright a program before letting it out of your sight? You'll need to consult a lawyer to get a "real" opinion. My personal opinion is that it's a waste of time. We're not talking about writing the successor to AppleWorks here. Just put "Copyright 1993 by Your Name - All Rights Reserved" and you'll be fine. No, you won't be able to sue anybody who "steals" your program but so what? Do you want a job as an Apple II programmer or do you want to litigate someone?

Unless your program is a Gee-Whiz-Way-Kewl kinda thing just release it freeware. That means you hold the copyright but people can use the program for free. If you think it's good enough to get money for, then releasing it shareware should be your second choice. First choice? See if someone will pay you a big chunk-o-money for it.

More on that later.

Okay, you have your program, you've tested it, and you've uploaded it to the online services. Now, sit back and wait for the job offers to flow in.

Six months later (or however long it takes you to figure out I was joking) get back to the computer and write another program. Do the same thing with the second as you did with the first. Oh, and by this time you should be logging onto GEnie at least every other day and becoming a "presence" in the programmer section (become a "presence" in the end user section too, if you have the money to spare).

How do you become someone on the online services? Attending the weekly RTCs (Real Time Conferences) is a good start. Answering questions that you see posted is good. And asking questions is also good. The smart Apple II programmer will do all of the above. The really smart Apple II programmer will come up with an idea that will make him "in demand" and garner him a free account on the service (ask me about A2 University the next time you see me <grin>).

Your online presence should be coming along nicely (be careful that you're a fun guy online and not an arrogant ass). Cultivating your physical presence is a little harder. KansasFest (The A2-Central Summer Conference) is the best way to meet all the hip Apple II programmers and publishers. The other Apple II/Mac trade shows on the coasts may help you somewhat. Just remember to meet and greet the important people in the industry while you're there. Standing off and drooling won't help you. Go up to Roger Wagner and introduce yourself. Say something nice about HyperStudio. Compliment his tie. The next thing you know you'll be living in his spare room and ghost-writing articles for him. Don't laugh. I've seen it work.

Time passes and people know you. They've typed at you online, they've downloaded your programs, you've helped people with problems, you've schmoozed with folks in person (optional), and now you're ready to make a little money. First you need a program that's worth something. A Widget Counter probably won't do it. That's a little too specialized. Let's say you've come up with a program that teaches the chords on a guitar and quizzes people on them. You could release that shareware and make $100 (if you're lucky). Or you could see if someone like, oh, I don't know, maybe, *Softdisk*, would like to buy it.

Stop! I can hear some of you muttering, "It's a setup! He works for *Softdisk* and he's just using this forum to drum up submissions!" My answer to that? "So what. Do you want to make money programming or do you want to mutter? We pay anywhere from $200 to $1200+ for a program."

Anyway, call Lee Golden at *Softdisk* (318-221-2173) and tell him you have a program you'd like to sell. Lee will tell you how to get it to us. The cool thing about selling a program to *Softdisk* is that we have an entire art department at our disposal. Which means that we can turn your stick figures into real people. We also have cool programmers (Bryan Pietrzak, Peter Rokitski, and me) who can help you with any problems that may crop up.

Your program may not be right for *Softdisk*, but there's also Roger Wagner Publishing, Quality Computers, Seven Hills, and a few more places that publish outside submissions. Send them in. They may not have a need for your program at that time, but you'll have gotten your name in front of them. And if you can program well, you're now in a position to ask them if they need any contract programming. The way to get contract jobs is to ask for contract jobs. Sending samples of your work (even if it's not a submission you'd like them to publish) will give them an idea of what you can do.

If you get a round of "No, thanks" to your offers to do contract programming, sit down and write another program. Send it to *Softdisk* as a submission or upload it to the online services. Keep programming. Keep your name in view. And a couple months later call the companies again and ask them if they need any contract programming done.

If you're any good at programming you'll start getting the jobs. If you're not any good you'll probably have already figured that out when everybody online dumps on your programs. If that happens, just keep programming. You'll get better.

What if you want to become a full-time Apple II programmer? The path is the same. But when asking about contract jobs be sure and mention that you're available for an in-house position--and willing to relocate. Keep your name in front of people (without being obnoxious) and keep programming. There aren't a lot of full-time Apple II programming jobs out there, but they are around.

Just in case you don't believe the advice in this article is practical, I offer myself as proof. I not only had no computer schooling, I didn't even go to high school. I got my first computer in 1984 and taught myself programming. I used GEnie to become a "someone." That led to freelance programming for *UpTime,* Roger Wagner, OKS, *Softdisk,* and others. It also led to Kansas City and working for Resource Central. And finally to Shreveport and *Softdisk.* I've been a full-time Apple II programmer for over three years now. Virtually every programming job I've had has been the result of a GEnie contact. If you don't have a modem yet, get one!

The system's in place. It worked for me and it can work for you - if you work it.

Final important note! You can get almost everything you need for programming the Apple II from Resource Central. If you have another outlet for programming materials, that's cool. I'm biased toward Resource Central because I used to work there and I know you can trust them. (The fact that Tom Weishaar has pictures of me cavorting in a Kansas City fountain has nothing to do with this endorsement. Honest!)

Apple Writer: Glossaries

by Ron Evry

Last time, I briefly mentioned the glossary file on *Apple Writer* called SPECIAL. It is a great introduction to what glossaries can do and if you have an Imagewriter, you should load it up to get a feel for them. First, press CONTROL-Q, and bring up the Additional Functions Menu. Then choose "E.(L)oad Glossary File".When prompted for the name, type SPECIAL, or if your prefix has been changed, type /AW2MASTER/SPECIAL and hit RETURN. These are the commands:

Open Apple-7	17 pitch type
Open Apple-5	15 pitch type
Open Apple-3	13.4 pitch type
Open Apple-2	12 pitch (Elite) type
Open Apple-0	10 pitch (Pica) type
Open Apple-e	Elite proportional type
Open Apple-p	Pica proportional type
Open Apple-B	Boldface ON
Open Apple-b	Boldface OFF
Open Apple-_	Starts underlining
Open Apple-	Stops underlining
Open Apple-h	Inserts a backspace

Try inserting these in a document, just to see how they look. Text produced with *Apple Writer* can be varied and interesting without going to much trouble at all. You can combine settings, so if you want 17 pitch bold underlined type in your file, go ahead. Remember that your printer will remain on whatever setting it was on last, even if you load in and print a new file that has no imbedded printer commands. You can always shut your printer down and turn it back on again to clear all of the commands.

Glossaries are incredibly useful. On the simplest level, you can use them to automatically type in frequently used words or phrases with a single keystroke. For example, say you are writing a review of *Star Trek III: The Search for Spock,* and you did not wish to type in

the title over and over again. To create an entry in a new glossary, you must first clear out the old glossary (unless you want to add these entries to SPECIAL). Press OPEN APPLE-G, then press the asterick (*). This will purge SPECIAL from memory. Then, press Open Apple-G again and press the question mark to DEFINE a new entry. You are now in what is called the Glossary Buffer. The screen will be clear, except for a prompt that says "Enter new definition:". Here you should type the key that will bring up the entry, followed by exactly what you want printed, then press RETURN. In this example we will use the lower case "s", so your entry will look like this:

sSTAR TREK III: The Search for Spock

Notice that there is no space after the "s". If you include a space in an entry, it will be printed. The entry key is case sensitive as well, so that you can have one entry for "s" and another for "S". You can use virtually any character for glossary entries, including numbers, punctuation marks and control characters. You may not use the left or right arrows, escape, tab, "?", "*", or these control characters: (A), (G), (H), (L), (M) (or RETURN), (U), (W) or (X).

Now that you have made your glossary entry, every time you press OPEN APPLE-S, "Star Trek III, The Search for Spock" will appear in your text instantly. Glossary entries are not limited to one line, either. Suppose you wanted to make an entry with your name and address in it. You simply insert a right bracket ("]") wherever you need a carriage return in a definition. Here is an example:

rRon Evry]2880 Cedar Crest Ct.]Woodbridge, VA 22192

Pressing OPEN APPLE-R would bring up my name and address, and print it over three lines. You can even imbed printer commands like so:

r].LM55]Ron Evry]2880 Cedar Crest Ct.]Woodbridge, VA 22192].LM0]

This sets the left margin at 55, prints my name and address, and then restores the margin to zero.

You can place control characters in a glossary entry as well. When you place a control character in a glossary, it will appear in inverse. So if you wanted to make OPEN APPLE-P bring up the Print/Program Commands menu, your glossary entry would look like this (with {p} representing an inverse "p"):

p{p}?]

You can also enter multiple commands in a single entry. Something like this:

4{Q}CMY.VALS]{P}PD0]{P}NP]

By hitting OPEN APPLE-4, you will then load up the Print/Program Value file called "MY.VALS.PRT", set the printer destination to the screen, then print the file on the screen in the preview mode.

By the way, if you want to actually include arrows as definition keys, or create entries that include cursor movements, CONTROL-U is a right arrow, CONTROL-H is a left arrow, CONTROL-J is a down arrow and CONTROL-K is an up arrow.

There are some limits to glossary entries, and there are workaround solutions to those limits. The maximum size of a glossary (the total number of characters of all the entries) is 2,048 characters. You are also limited to 99 entries in a glossary. The most characters allowed in a single definition created in the glossary buffer is 128. However, you can make glossary entries that call up other glossary entries, so that they will string together. Do this by inserting a CONTROL-G in an entry, followed by the entry definition letter you wish to access. For example, if I added this definition to the list:

a{G}r]{G}s

my name and address would appear, followed by a carriage return, then "Star Trek III: The Search for Spock." whenever I pressed OPEN APPLE-A. You can call on a whole chain of definitions if you like, but

be careful not to nest them more than eight deep. You also can create glossary entries that will load up new glossary files. For instance, I have been using a glossary file I named "MAN.GLOS" while writing this article. It contains frequently used words like *"Apple Writer"* and "delimiter". It also has an entry that loads up the glossary file "SPECIAL" whenever I need it, and of course, I have put in an entry there to get me back as well. The only limit to the number of glossary files you can make is the amount of disk space you have. And if you do not have a hard drive, you can always swap disks.

It is also possible to create glossaries in the text buffer. All glossary files are text files arranged in a special manner, and loaded in a special way. The advantage of creating glossary files here is that individual entries can be much longer. In fact you can make a single glossary entry in the text buffer 2,048 characters long, but you would have room for no other entries.

There are some differences — if you wish to enter a control character in your definition, you must embed it with CONTROL-V before and after. Also, this procedure will not work successfully with printer commands that need to be embedded, but not executed. If you really need to do this, you can use a neat little file on the *Apple Writer* disk called CONTROL-V. Next time around, I will try to explain how it is used. In the meantime, if you want to create glossary printer commands, you can easily do it one at a time in the glossary buffer.

If you are creating your glossary in the text buffer, you would save it the same way as any text file with a CONTROL-S. But you may want to give the file a ".GLOS" suffix to make it easier to find.

If you have added entries to a glossary from the glossary buffer, you will probably want to save it. In this instance bring up the Additional Functions menu with CONTROL-Q and select option "F". At the prompt, type in your glossary name, including the suffix, if you want it. If you are replacing an earlier version already on disk, *Apple Writer* will ask if you want to delete the old version.

In the next, and final, stop of our *Apple Writer* tour, we will explore glossaries a bit more and open up the most powerful toolkit ever put into a word-processing package: WPL, the built in programming language that does everything except walk your dog. We will also find out where to get some nifty patches for *Apple Writer* that will enable you to print fancy fonts on an Imagewriter or Laser Printer, include pictures in your printouts, and lots more.

Miscellanea

Seven Hills Software (2310 Oxford Road Tallahassee, FL 32304-3930, 904-575-0566) unleashed a well-kept secret last week with the announcement of *The Manager*, a MultiFinder for the Apple IIgs. We haven't received a copy in the office yet, but according to the press release, *The Manager* will allow multiple applications to be opened simultaneously. Moving between applications will be as easy as clicking in the desired window. Seven Hills states that *The Manager* requires System 6.0 and 2 megs of memory but advises 4 megs for greatest efficiency. A hard drive is not mandatory but is recommended for optimal response time. It will be extremely interesting to compare this product with Procyon's *Switch-It!* which was announced last month. We hope to do an in-depth comparison in the April issue. Stay tuned.

You may have heard about the addictive, amusing and educational game called Lemmings which is available for MS-DOS and the Macintosh. It's the kind of game that makes you think in ways that your brain probably isn't used to thinking and it's fun at the same time. Word has it that if the programmers get enough requests, they will seriously consider doing an Apple IIgs version. Write to Psygnosis (29 Saint Mary's Court, Brookline, MA 02146) and tell them you want Lemmings gs.

This is not really Apple II specific but I figure that the poets among us will be interested to learn that the National Library of

Poetry will be giving away $12,000 worth of prizes to 250 winners of the North American Open Poetry Contest. The contest is open to anyone and entry is free. To enter, send *one* original poem, any subject, any style, to the National Library of Poetry, 11419 Cronridge Drive, P.O. Box 704-XE, Owens Mills, MD 21117. The poem should be no longer than 20 lines, and the poet's name and address should appear at the top of the page. Entries must be postmarked by March 31, 1993. A new contest will open April 1, 1993.

Folks who subscribe to GEnie can take advantage of a new programming course offered by A2Pro. A2 University is starting a new 6 week course on resources. The course will be taught by Marc Wolfgram of Lunar Productions and will be aimed at both programmers and users. If you've been curious about resources and want to learn more about them, this is the course you've been waiting for. Live RTC's will be held on Monday nights starting February 15 and the lessons may be downloaded from A2Pro's library #16. Participation in the RTC's are not required but Marc will be there to answer any questions and to give you guidance. An outline of the course follows

Week 1: What are resources? Comparisons and benefits.
Week 2: A technical sidetrack into file structures.
Week 3: Resources and tools, part one.
Week 4: Resources and tools, part two.
Week 5: System 6, the Finder and other goodies.
Week 6: Programs for resources, a survey of what's out there.

GEnie sign-up information can be found in our monthly catalog or by calling 800-638-9636.

SFPD

No, it's not the San Francisco Police Department, it's a new and recurring column designed to keep you informed of the various shareware, freeware and public domain programs for the Apple II. With fewer and fewer commercial developers spending their talents and resources on the Apple II, we have come to depend on the loyal but few commercial developers left and the SFPD market for programs to entertain us, to enrich our lives and to increase our productivity. Unless you're a member of an active user group or a modem owner who is a frequent visitor to the major online services, you're likely to miss out on most of the SFPD offerings. The people who write these programs do so, not for commercial gain, but because they enjoy the challenge and still find Apple IIs to be their computer of choice. If you use the programs described in this or future columns , please remember to pay the shareware fees when applicable. There's little hope of seeing future programs if the efforts of the programmers who do them go unrewarded. Even if a program is freeware, it wouldn't hurt to sent a note (or some other gesture of appreciation) to the developer as a way of saying, "Thanks!, I like what you're doing."

ShadowWrite **is an incredible NDA (new desk accessory) for the Apple IIgs that's as close to a full fledged word processor as I've seen.** Not only can you work on up to eight documents at once, but they can be Teach, text or SRC files. In this space, I will attempt to relay the other unique features of *ShadowWrite*.

ShadowWrite's yellow menu bar gives you options for five pull down menus: Apple, File, Edit, Layout and Document. In addition to the About box, you'll find a Preferences menu here. Preferences include Menu Key Equivalents, Auto I-Beam Cursor, Give others First Shot, Default Font, Default Text Font, and Hide Dimmed Files.

The most exciting item in the File menu is the option to load any type of file with the Load File option. Coupled with the Modify text option in the Edit menu, you are able to import Macintosh (or Apple II) word processor files of any format, strip the control characters via the Modify Text item and have a readable and editable docu

ment with which to work. You can print your file from *ShadowWrite* or (if you don't Give Others First Shot) you can highlight a file supported by it in the Finder and Print from the Finder's File menu!

The Edit Menu allows you to Find/Replace (Find Next, Go To Start, Ignore Case, Replace or Replace all), and Modify Text. The Modify Text options include specifying whether you want to modify the whole document or just a selection, Clear Highbits, Strip Controls and Convert Text. The latter command translates 8-bit characters into text that can be displayed on the Apple IIgs screen. The last item in this menu allows you to encode/decode the text with a keyword. Great to protect your data from prying eyes or shareware authors could use this feature to protect their docs until shareware fees are paid! <grin>. This item, as well as the Strip Control option, are touted as being exclusive to *ShadowWrite*.

The Layout Menu allows you to Choose Font (style and size too), Choose Color and Set Ruler. The Document Menu gives you a list of the open documents and Text Info (Status, Characters, Lines, Styles, Used Memory, and Selection, which shows size of highlighted text). A Compact Memory button is also available from this dialog box. Since it's an Apple IIgs specific application, you can make use of *Pointless* and TrueType fonts for printing.

That's not all; it's got keyboard equivalents for just about everything, it's not crashed my system, and it's freeware, folks! This great little NDA is brought to you by Andre Horstmann (Hohenweg 3d, Ch-6300 Zug, Switzerland) and Bright Software (P.O. Box 18,

4153 Reinach 2, Switzerland). If you use it and find it half as useful as I do, drop Andre a line and tell him so.

FloorTiles 2.0, **by Karl Bunker, is a steal at the $10.00 shareware fee.** It's a little distraction that easily becomes addictive, exercises your brain and can be enjoyed by family members of all ages. The playing board is a white floor-like surface upon which colored tiles are placed. Each tile is made up of four randomly placed colors and is controlled by the mouse. Tiles can be rotated with the spacebar and placed anywhere on the board by clicking the mouse button. Points are scored for each tile that is placed on the floor; the more quickly you place a tile, the more points you will receive for it. The game is over when all the white spaces are filled, so in order to keep playing and to achieve a high score you need to match the colors of three sides of the square to the squares adjacent to it. That makes the square disappear. If you're really good and match all four sides, the tile and the ones surrounding it will vanish as well. When a tile disappears, the adjacent tiles rotate 90 degrees, just to keep you on your toes.

There is an option for choosing a large board (9 X 11) or a small board (5 X 9). You can save a game in progress, pause it, and view high scores for both board sizes. This is definitely a shareware game of commercial quality.--edr

Ask (or tell) Uncle DOS

Power II be

I own an Apple IIgs (ROM 03) with an Apple 3.5 drive, a Chinook CT40 hard drive and an ImageWriter II printer that will continue to comprise my desktop setup for the foreseeable future. However, I am planning to buy a PowerBook to use on the road, mainly for word processing, spreadsheets and communications. Upon returning home, I would want to upload the latest PowerBook files to the IIgs hard drive and/or print them. For those simple needs, even a 145 would be adequate but I was surprised to find out that I could buy the more advanced 4/80 Duo 210 for $400 less than the 4/80 145 (and even $100 less than the 4/40 145). However, this is true only as long as I don't have to buy an external floppy drive and some type of dock to connect it to the Duo 210 (the 145 includes an internal drive). I am assuming that I would use an internal modem in all cases.

I have no experience at all with AppleShare, but Dennis Doms's excellent article on "Apple-Share Connectivity" in the May '92 *A2-Central* suggested to me that I could connect the IIgs, the ImageWriter and the PowerBook into a

LocalTalk network, using the PowerBook as a miniserver. In this way, the PowerBook would need only a LocalTalk port (and an RJ11 connector for the modem) that all models, including the 210, provide. The network, including the AppleTalk Option Board for the ImageWriter, would be quite a bit cheaper than an external drive plus even the simplest dock (the Floppy Adapter), thus preserving about half of the 210 cost advantage over the 4/80 145.

My question then is, how realistic is this approach? I feel confident that it could meet my routine needs for transfering text files, but how about installing software, and particularly system software? Can the IIgs on the network be used for that, or would I have to "borrow" another Mac to do it? Or would it be a more robust solution to spend a little more for an external drive and a dock, and use floppies for file transfer as well as software installation? That would still be no more expensive than the 4/80 145. Either way, I feel the 210 would give me more flexibility if I later discover more applications for the PowerBook.

Any advice will be truly appreciated.

Guy Desloovere
Cambridge, Mass.

On the networking side (and transferring files) everything should be fine. AppleTalk isn't the fastest way to move files, but I've found it fast enough. (Not much worse than floppies.)

Regarding adding downloaded software: Probably not too bad. Assuming you have the requisite items already on the Duo, items that you download (from the Mac RT, for example) could be downloaded on the IIgs, taken to the Mac, and expanded (or, I guess, even downloaded on the Mac if you have a modem for it). To do this, you'd need at least UnstuffIt Deluxe (freeware; unpacks most Mac archive formats) and a Mac terminal program with MacBinary

support (I use ZTerm, which is $30 shareware) or (if you plan to download on the IIgs) a MacBinary extractor (I use "MacBinary", also freeware in the Mac RT on GEnie).

Adding software from disks could be tricky.

*If the software is available to copy as **files** from the Mac disk through the IIgs (networked) to the Mac, it should work. I haven't tried doing this specifically, but I have been able to move Mac files to and from the IIgs over a network and have it work. Obviously I haven't tried every type of Mac file, but assuming all the filetype/creator assignments are preserved accurately by the AppleShare and HFS FSTs, it should work.*

The problem: many Mac items come on disks which require running an Installer to add to your disk.

On disks images you download (say, system software from the Mac RT): most of these will probably be in DiskCopy (Mac program) format. There is a Mac Control Panel called "MountImage" (also in the Mac RT) that will allow you to "mount" these DiskCopy images (up to 8 in the most recent version of MountImage) on the Mac desktop and then use them as if they were floppies. I've had to use this technique on an emergency basis, and sometimes it works. With some disk images and installers, it doesn't.

If it's a commercial disk, you may be somewhat stymied since you'd have to be able to create the DiskCopy image before you could copy it to the Mac and mount it with MountImage. Legalities aside; let's assume this falls into the definition of "allowed backup" and you can create a DiskCopy image of a program you own in order to install it on your computer.

I believe the DiskCopy format is defined in t

he latest set of File Type Notes. I don't know if there is a program to support it (maybe GS+'s disk duplication program can create DiskCopy image files?); if there is, that would be a way to get the disk image into a form you could try to install. I say "try", because (like I said) it doesn't always work, based on the tolerance of the installation program.

My opinion: don't try it this way unless you have at least **occasional** access to a Mac with a floppy. Even so, I can see some sort of knee-jerk problem down the line where you could be left in a situation where you couldn't get to the Duo hard disk (say, a crash) and the house of cards would fall unless you could "borrow" a Duo floppy setup. If you can find a system with a drive for $300-$400 more, it may be worth it in saved aggrivation.

RE PowerBooks in general: unless you plan to live with the LCD screen forever, you may want to look at a 160 (with video out and a floppy). Adding video to a 100/140/145/170 is expensive. Other types of expansion may be greater (RAM) or less expensive (high-speed internal modem) too. (That is,extra money now may save you some down the road.)

The biggest advantage I can see to a Duo is if you can swing the desktop dock. The Duo is more portable (no internal floppy, after all) in size, but again expansion costs catch up with you (sometimes they get you coming, sometimes they get you going :).--DJD

Auto off

Back in November I wrote to you regarding crashes I was encountering using GSHK 1.1. It took a while to find the cause because of all the stuff I have loaded. System 6.0 and Bootmaster from Q-Labs makes the process of isolating various INITs, CDEVs, NDAs, and CDAs pretty easy, although it still takes time to work through all of it.

One discovery usually triggers other; however, the trails seem to end as follows: AutoMenus 1.2 causes GSHK 1.1 to crash and creates problems with with Platinum Paint 2.0 menu extensions. I would suggest just deactivating the init in the control panel or under Item Info and wait until Jay Krell fixes it. It's a very useful init.

MenuTime 3.33 cannot deactivate its screening when working in Platinum Paint 2.0, 320 mode (covers 'Window' in menu bar).

John W. Evans
Ashland, Wisc.

More *Apple Writer* tips

I have been using Apple Writer since I got my Apple IIe two months after it was released. I used the DOS 3.3 version for years and then moved up to the ProDOS version when it was available. This program has it, hands down over AppleWorks (word processing). Ok, AppleWorks has a few more features built in, but Apple Writer is much more compact and much, much faster. Try doing a global search and replace with each program on a substantial file and you'll see the difference. On one file that I tried to do a global search and replace, AppleWorks

took over an hour and was only half way finished; Apple Writer did the whole job in just under 10 minutes! Between WPL and the glossary, there's not much you can't really do! Whenever I have a choice between AppleWorks and Apple Writer, Apple Writer wins every time!

Some tips—in the first segment of Apple Writer: A Guided Tour, Mr. Evry mentioned one quick way to save a file (after that first time that it is saved). There is an even quicker way if you are not changing the file name.

When you press control-S(ave), you are given the prompt along with the last used file name. If you want to use that same name, just press the "=" key and then Return and the file will automatically be saved to the same disk as before using the same file name.

I have also prepared a "Startup" file, that 'sets up' Apple Writer the way I want it with my most used glossary (macro) keys. I have also adapted one of the WPL files from the Apple Writer disk to number paragraphs (or anything else that I may need numbered). A gentleman in our computer club claims to have written several disks full of glossary entries!

I have run into some problem areas with Apple Writer and usually try to ignore them, if I can. I always have a problem getting Apple Writer to format a disk so I never use that function.

The one wish item that I have is the ability to use more of my Apple's memory than just the stock 128K. I have a one meg board (made by AST, which follows the Apple memory format and not the Applied Engineering format). Is there a patch that could be made so that I can have access to more memory? I use version 2.0 instead of 2.1—for some reason (now too obscure to remember), I had problems with 2.1 and went back to the previous version.

Howard Bartnof
Van Nuys, Calif.

BASIC conflict

I'm writing in response to the Jan 1993 issue of A2-Central. If A2-Central were in competition with magazines like InCider/A+, then perhaps a dominating concern might be to somehow obtain professional, well-written articles that are smooth and interesting to read, above all other concerns. But A2-Central is not considered to be a clone of other computer magazines. Its readers subscribe to a flimsy-paper, newsletter-like magazine at prices comparable to the "slicks" because it has been unique. A reader could believe what it says, because everything was written by people whose income did not depend on the product they were writing about. The article on BASIC: The People's Language, in the January issue, was very good, interesting, informative, and I even agree with most of what it said. But whatever it said that I didn't know from experience to be true, I would take with a whole pile of salt. Why? Because it was written by the people whose income derives from selling the products being discussed. Sure, they know more abut those products than anyone else. So what? It makes absolutely no difference whether a

presentation contains the truth, the whole truth and nothing but the truth, if no one believes it. (You doubt?—Talk to anyone who ever reported a UFO sighting.) So that article, well-done or not, conflicts with the image of A2-Central has built for itself over many years. And of course, the article was biased in that its scope was limited to specific products of direct concern to the authors.

In spite of the fact that I have used MD BASIC for years and am very happy with it, and that I intend to acquire Micol BASIC, I have also used Blankenship Basic, Probasic, LL BAsic, Tiny Basic, and a couple of others that run on Apple II's. I mention this to support my comment on scope.

One reason I was not among those who tore their hair out at the demise of A+ was my annoyance that a significant percentage of its "reviews" were plain and simple advertisements, derived primarily from the backs of the software boxes. We live in the world of docutisement, when TV marketing is loosely camouflaged as information. Sometimes there even is a little information thrown in for verisimilitude. The bottom line is, I hate to see A2-Central, of all information sources, present a feature article, no matter how authoritative and well-written, that is simply an advertisement with some history and discussion thrown in. It's a good way to kill your credibility. And credibility is what sells your journal.

Phil Albro
Cary, N.C.

More on MonoSpaced Fonts

In the January issue of A2-Central you asked for info about TrueType Monospaced fonts. There are 3 monospaced fonts on the Pointless disk. They are Courier, Courier Bold, and Monaco. In my collection I also have a version of IBM Klone that is TrueType. I think I downloaded it from GEnie, but if I am correct it was part of one of the NAUG TT font disks.

Bruce Milyko
Fairfax, VA

Couldn't help but notice that your response to Don Zahniser left out one bit of information that I'm surprised neither you nor Marty Knight noticed. I just bought your True Type font package by II Productive and it contains the IBM Klone font on disk #10.

Randy Chevier
Tulsa, Okla.

The font collection can be ordered from us or they can be downloaded from Apple II bulletin boards world wide.—edr

People helping people

Here are a couple of tips I've discovered. I don't know if either of them are common knowledge but I've not run across them elsewhere. For what they're worth.....

As every Apple IIGS owner with System 6.0 knows, to get into the Finder after setting Boot-Start to another startup program, you have to

hold down the Command key while booting. This can be a real bore especially if the Boot icon doesn't' come up until well into the booting process. However, if you press Command and another key, then you can go make breakfast or whatever while you IIGs boots into the Finder. But if you use Kangaroo/Hierarchy, it seems to swallow the keystrokes as they install themselves. Just wait until after these two icons come up, then type Command-(another key). These are the only two items I know that swallow keypresses but there may be others. Also, if the second key you type happens to be a menu shortcut key that is active at the moment the Finder opens (before it puts any icons on the desktop), that action will be taken immediately. For instance, Command-Q brings up the Shutdown dialog. Command-spacebar works fine as the second key.

As for the second tip, I have suffered with numerous Fatal error 911 crashes since getting System 6. I mentioned this in an earlier letter which you printed. I believe I may have finally found the source of the problem and it's not System 6 at all. The IIGs I have been using was set to run on 50 hertz although the current where I am now living alternates at 60 hertz. Reason for my mistake: Inexplicably the northern part of Japan runs on 50 hertz while the southern half boogies at 60 hertz, although voltage is the same 100 volts country-wide. About

the time System 6 arrived, I bought a second Apple IIGs from a friend who was selling out to the Macintosh. He lives near Tokyo, north of the 50 hertz line, while I live in Hiroshima, south of it.

Because the monitor didn't exhibit any overt misbehavior, I never think to switch the hertz. (The background area of the monitor display was slightly scrunched toward the tip of the screen but not enough to raise my suspicion.) I suspect that all TVs and monitors manufactured in Japan are able to manage both 50 and 60 hertz, for obvious reasons. The AppleColor monitor is made by Mitsubishi. However, since one kind of IIGs interrupt uses the Vertical Blanking Signal for timing, it stands to reason that the wrong hertz setting could cause more problems than just the vertical rolling of the monitor which the manual describes. (Error 911 indicates a Desktop Bus interrupt problem.) It's only been a few days since I made the switch but I've even reinstalled some the the the DA's I had thought were the source of the problem and so far, so good. Cross your fingers.

To reset the hertz, hold down the option key when booting and choose from the resulting menu. Some Apple IIGs owners probably have never seen this screen, nor would have a need to. Be warned that resetting hertz also sets all your Control Panel settings to default, so "If it works, don't fix it." Furthermore, the hertz setting is apparently saved in battery RAM with everything else. When I tried to reset my Control Panel settings with Lithium Grease, I was back to the 50 hertz mode. No short cuts on this one, I think. You have to change all your Control Panel settings, one by one.

Bill Robbins
Hiroshima, Japan

A2-Cental's masthead proclaims it to be "a journal and exchange of Apple II discoveries." This letter certainly conveys that spirit. Thanks, Bill. On another note, Fatal System Error 911 can also signal an overheated computer. –edr

Bootless

I have been trying to use my /RAM5 as a startup disk, with no success. Here's what I've done:

1. Created an 800K RAM disk through the Control Panel. Rebooted.
2. Copied the files PRODOS (2.01) and BASIC.SYSTEM to /RAM5.
3. Set startup slot to "RAMDisk".

When I reboot, I had expected to boot into /RAM5. Instead I got the sliding apple and "Check Startup Device." error message. What am I doing wrong? It is possible to boot into /RAM5, isn't it?

Dan Crutcher
Louisville, KY.

Simply creating the RAM disk is not enough, you also have to initialize it. From the Finder, highlight the RAM disk, then pull down the Disk menu and choose the initialize option. Then copy the appropriate system files to the RAM disk. And away we go.–edr