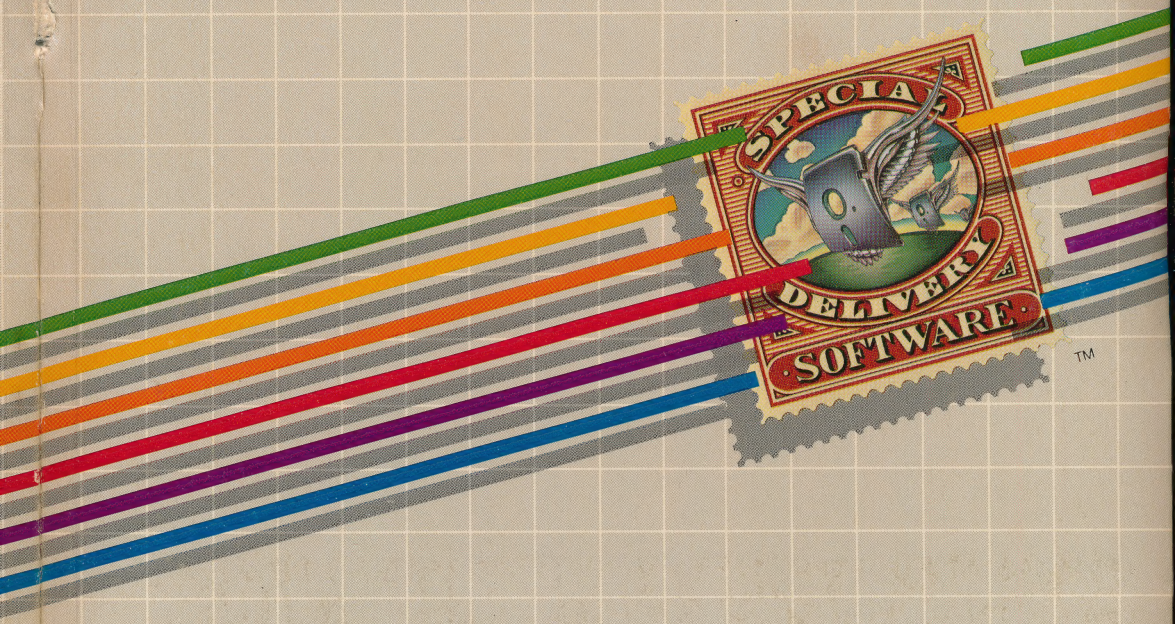
A P P L E



C2B0001

$75.00

SALE

$25.00

apple computer inc.

PASCAL ANIMATION TOOLS

Written By: CHARLIE KELLNER

In conjunction with Apple Computer, Inc.

Art and typesetting prepared by H. B. Kappelhof and Associates, 396 Martin Avenue, Santa Clara, CA 95050 • (408) 727-2053 • Job No

NOTICE

Apple Computer Inc. reserves the right to make improvements in the product described in this manual at any time and without notice.

DISCLAIMER OF ALL WARRANTIES AND LIABILITY

APPLE COMPUTER INC. MAKES NO WARRANTIES, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS MANUAL OR WITH RESPECT TO THE SOFTWARE DESCRIBED IN THIS MANUAL, ITS QUALITY, PERFORMANCE, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. APPLE COMPUTER INC. SOFTWARE IS SOLD OR LICENSED "AS IS." THE ENTIRE RISK AS TO ITS QUALITY AND PERFORMANCE IS WITH THE BUYER. SHOULD THE PROGRAMS PROVE DEFECTIVE FOLLOWING THEIR PURCHASE, THE BUYER (AND NOT APPLE COMPUTER INC., THEIR DISTRIBUTOR, OR THEIR RETAILER) ASSUMES THE ENTIRE COST OF ALL NECESSARY SERVICING, REPAIR, OR CORRECTION, AND ANY INCIDENTAL OR CONSEQUENTIAL DAMAGES. IN NO EVENT WILL APPLE COMPUTER INC. BE LIABLE FOR DIRECT, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT IN THE SOFTWARE, EVEN IF APPLE COMPUTER INC. HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF IMPLIED WARRANTIES OR LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE LIMITATION OR EXCLUSION MAY NOT APPLY TO YOU.

10260 Bandley Drive
Cupertino, California 95014
(408) 996-1010

Table of Contents

Overview

This manual describes the facilities provided on the Pascal Animation Tools Diskettes (Volumes 1 and 2). In addition to Animation Tools, we have also provided additional tools from our Software Engineering Lab for the Pascal environment including graphics tablet interface, and a communications card interface. In order to provide these additional Pascal Tools at minimum cost and delay, they are released in source code form which Pascal programmers will find to be nearly self-documenting, enjoyable to study and adapt to their own projects. There are several source code files in the total product. Therefore, this manual only briefly defines the functions of such files, and is not intended to be a tutorial style document.

I. Introduction

Since the "early days" of computing, the primary purpose of graphic displays, like the computer itself, has been mathematical in nature. The first computer displays, of course, were limited to a few rows of miniature light bulbs. These tiny lights were forever flashing on and off in rapid-fire patterns, which mainly served to let the computer operator know the machine was still working. Oddly enough, this form of information display continues to be popular on many large business computers.

Almost immediately, however, second-generation computers began to employ cathode-ray tubes (CRT's) as display devices, providing several thousand individual points which could be turned on and off. These points, like the light bulbs, were simple binary representations of the contents of the computer's memory. By carefully altering the memory contents, computer scientists soon discovered that a recognizable picture could be made to appear on the CRT display. Thus was born the lively art of computer graphics.

Today's computers use a variety of graphic displays, many of which offer considerably greater resolution than a home television set. Most high resolution graphic terminals, however, tend to be very costly (and require frequent "tune-ups"), and so are still mostly used for scientific purposes.

But along the way, a curious thing happened: the personal computer. Based on a new bit of technological wizardry called the "microprocessor", micro-computers like the Apple II turned a mathematical tool into an art form. Coupled to a standard color television set, one of the personal computer's most interesting features is (naturally) color graphics, and one of the most popular applications of today's computer graphics is animation.

This manual assumes that your Apple II system is correctly set up. If you're not sure that the system is ready to go, see Appendix A.

II. Animation Techniques

In reality, there are nearly as many different ways to animate a picture as there are computers. Most of the time, though, the animation technique chosen for a particular application will depend on the type of display terminal used. For example, one might expect line-oriented animation to be used on a vector terminal (which automatically draws straight lines between the plotted end-points). Anything else would probably be a waste of time.

The Apple II, however, uses a somewhat less complicated technique called a "bit map" display. This means that each individual bit of the computer's memory controls one point on the display screen. Each horizontal display line represents hundreds of adjacent memory locations. In the Apple's high resolution graphics mode, for example, there are 280 points per line, with 192 lines per frame. Each point can be one of six available colors. The image produced by this technique has the advantage that it is compatible with a standard television receiver.

One type of animation that has proven to be very successful with the Apple II's bit map display is called "cel animation". This technique began many years ago in the motion picture industry with the animated cartoon. In an animated motion picture, thousands of individual drawings called "cels" are photographed on movie film, and later shown at the rate of 24 frames per second. Each frame may consist of from one to several dozen cels, providing both subject and background detail.

In the simplest sense, then, cel animation can be accomplished by changing the computer's entire display memory contents 24 times (or so) per second. Actually, good animation is possible with as few as ten frames per second. This can easily be done with the Apple's low resolution screen which occupies only one "K", or 1,024 bytes of memory (each byte is eight bits). In fact, as many as 47 successive frames of a five-second Lo-Res "movie" can all be loaded into a 48K Apple's memory, and simply moved into the display area at regular intervals. This has been done experimentally with digitized images from a television camera with surprisingly good results.

For more realistic animation, it is desirable to use the (28 times) greater detail of the Apple's high resolution graphics screen. This presents a significant problem in data transfer, however, since the Hi-Res screen occupies 8K, or 8,192 bytes of memory. At a clock rate of about one million cycles per second, just moving 8K of data from one place to another takes the

Apple's microprocessor about one twentieth of a second. This might be acceptable if all of the frames were in memory at once, but the same five-second movie would now take up over 300,000 bytes of memory!

The solution to this dilemma is simply to move a smaller part of the Hi-Res screen memory instead of the entire 8K. In fact, it is often the case in motion pictures that the subject(s) move while the background remains stationary. Thus, with subjects which occupy less than a tenth of the screen area, and careful programming, it is possible for the Apple to animate several figures at once, at up to 30 frames per second. The only trick, up to now, has been the "careful programming" part.

III. Tools of the Trade

As mentioned previously, the best general procedure for Hi-Res cel animation is to rapidly change small blocks of the display screen under program control. Of course, it is always possible to write a short assembly language subroutine each time you want to animate something, but there is a much more convenient solution.

Apple Computer, Inc. now offers three environments for high-speed, high resolution animation on the Apple II: The DOS Toolkit, Apple PILOT, and the Pascal volume HIRES1. The special environments provided by these program packages have one important feature in common: The ability to print high-speed text on the high resolution graphics screen. This may not sound like very much of an advantage, until you consider that the Hi-Res text consists of upper and lower case characters as well as various special fonts, including pre-defined graphic shapes.

In addition to this capability, each of these packages also includes a graphic character set editor for creating special fonts. This combination of tools, in addition to its many other uses, is ideal for cel animation. The remainder of this manual will describe specific examples for use with HIRES1, but the techniques are applicable to all environments. For more Applesoft and Integer BASIC examples, please refer to the DOS Toolkit Reference Manual.

IV. Building Character Fonts

Each printable character in a font consists of eight bytes which are displayed as a graphic block seven dots wide by eight dots tall. Under control of the character set editor, any desired image may be defined as a group of printable characters, each made up of 7 by 8 dots. Images of up to 7 by 6 characters (49 by 48 dots) may be drawn directly on the graphic editing grid.

Larger subjects may be built in sections, and later assembled into complete pictures.

The Hi-Res Character Generator is a set of assembly-language subroutines for displaying text on the high-resolution graphics screen. It serves the same function as the Apple's hardware character generator (described in the section, The Video Generator, in the Apple II Reference Manual). It turns a series of ASCII character codes into a bit-by-bit map of the Apple screen. Each one-byte ASCII code produces a 7-wide-by-8-high dot image on the high-resolution screen, covering an area equal to two low-resolution blocks stacked vertically. Each dot will be on (white) or off (black), and the dots in each line of a character image can be shifted sideways by half a dot to produce smoother contours. The Hi-Res Character Generator can produce the same character set as the hardware character generator and many others besides. For example: Foreign alphabets and decorative alphabets, both with upper and lower case; and graphic alphabets for animation which carve up a large image into 7x8 blocks which can be displayed anywhere on the screen.

To do this, simply execute the file CHAREDIT.CODE, and press the carriage return key when asked for a character set name to define a new font. When the graphic editing screen appears, type in a block of characters for the first animated frame as you would like it to appear when it is printed. To begin with, the standard ASCII definition of each character will appear in the "Letters" box as it is typed. The "Graphic" box will remain blank until all of the characters have been entered, and the editing process begins. Also note that within the editor, an explanation of the commands will appear if you type "?".

Depending on the editor you are using, you may then sketch the figure to be animated directly on the screen, using the Apple's keyboard, game paddles, and/or graphics tablet. The drawing will appear simultaneously in the "Graphic" box, and on the larger editing grid, expanded four-to-one. When you have drawn the first frame to your satisfaction, make a note of the block of characters you used to define it, and save the character set by quitting the editor and typing its name.

To define the next frame, simply restart the editor, and edit the previously named character set. When the editing screen appears, type in a new block of characters and sketch the object in its second frame of animation. If some parts of the picture are similar to the first frame, you may economize on new characters and speed up the editing process by re-using the previous ones. The same characters may be used as many times as desired in a single animation sequence. Of course, it isn't absolutely necessary to save the character set after each frame, but it's a good idea if you don't want to risk losing your pictures accidentally.

V. Bringing It to Life

The graphic screen drivers provided with the previously mentioned Hi-Res environments are designed to respond to a wide variety of special control characters, in addition to the usual ASCII standards like backspace and carriage return. Because of this, all of the screen control sequences required for cel animation can be embedded in a string, along with the characters which make up the picture.

Most of the decisions involved in building animated string sequences have been greatly simplified by the Pascal ANIMATION library unit. The ANIMATION unit provides five procedures for flexibility and convenience: BLOCK, LOADFONT, USEFONT, STDFONT and ANIMATE.

To define a frame in Pascal, use the library procedure BLOCK (Frame, Image, Motion). The variable "Frame" represents your final result string, and should be declared as type "picture" for best results. "Image" is a literal which describes the block of characters used to build the picture, with a slash (/) between each row. "Motion" is one of the directives [stay, up, right, down, left, upright, downright, downleft, upleft], indicating the action to be taken after the block is printed.

The procedure LOADFONT (Font, Name) is used only once for each character set at the start of your program. The variable "Font" is the character set itself, which should be declared as type "Font". "Name" is a literal string which describes the actual name of the font file on the diskette, including the volume name, if desired.

Once a font is loaded from disk, it can be "switched on" instantly, by the procedure USEFONT (Font). The specified font will then be used for all future printing, until another USEFONT is declared. The procedure STDFONT will cause the standard ASCII character set to be used again. By the way, it's always a good idea to do a STDFONT before exiting your program; otherwise, the Pascal system messages are likely to look a little strange.

When you have invoked the proper font, your block images can be animated by simply using the procedure ANIMATE (Block, X, Y, Time). "Block" is of course the previously defined picture string. The variables X and Y specify the figure's current position on the screen (in text coordinates, 0-39 by 0-23). These must be set when the figure is first placed on the screen, but will be maintained automatically thereafter by the ANIMATE procedure. "Time" is a delay factor between frames, which may range from 0 (as fast as possible) to 32767 (about one frame every ten seconds).

An excellent example of the use of these procedures may be found in the programs "Maxdemo" and "Movie" on the HIRES1 diskette.

One last word: Despite its apparent simplicity, cel animation has proven to be one of the most practical techniques used in the motion picture industry, in terms of both time savings and results. The specialized Hi-Res screen drivers now available from Apple Computer, Inc. make true animation a reality, with printing speeds of up to 2,000 characters per second. Their potential for creative use should not be underestimated. In the immortal words of Boris Karloff, "Have fun - Experiment with it!"

VI. Fonts on Diskette

The fonts MAXWELL, HORSE (1 & 2), and BIRD (1 & 2) are included on this diskette for use with the Animation demo programs. The character sequences which make up the various animated figures can be easily seen by listing the text files MAXDEMO, HORSE, and BIRD. All of these fonts were designed using the CHARTAB program.

VII. Pascal Source Programs

STR-FP and STR-INT are short programs which demonstrate how to convert a string to numeric form (either real [FP] or integer [INT]). The programs each contain one procedure which does the conversion, and a short main program which shows how to use it. STR-FP uses the Transcendental library unit LOG procedure for overflow/underflow checking, but could easily be converted to do simpler limit checking without it, as demonstrated in STR-INT.

TABLINK is a set of assembly procedures for linkage to the Apple Graphics Tablet. They seem to work quite well, but represent only a small portion of the capabilities of the tablet, namely: Initialize, Read (X, Y and Pen Status), and Scale (X, Y). The programs TABLINE and TABDRAW have been included to demonstrate their proper use. TABLINE lets the user draw straight "rubber-band" lines, and TABDRAW is for continuous drawing, dot by dot. Neither program is intended for actual use as a picture editor, but rather just to show how it's done.

ANIMATION is a set of Pascal procedures for doing character cel animation under control of HIRES1. The procedures are also contained in the system library on HIRES1, but the actual source code will be extremely helpful for those who want to go beyond simple block figures and explore the true potential of this technique. These same procedures are used by the programs MAXDEMO, HORSE, TROT, RACES, MOVIE and BIRD, which are also contained on this diskette for reference. Besides, they're fun!

ALFTONE consists of several assembly procedures for communicating with the ALF music boards from Pascal. Again, only the basic

functions are represented: Initialize and Set (Slot, Oscillator Number, Period and Volume). Actually, this is all the boards are capable of doing so it should prove sufficient for using ALF boards.

COMREADY is an assembly routine for simply checking whether the communications card in slot #2 has received a character. It is used in exactly the same manner as KEYPRESS, and can be helpful in bidirectional (full duplex) communication from Pascal.

CHARTAB is a program provided so that those people with a Graphics Tablet may use it to define their alternate character sets. It serves exactly the same purpose as CHAREDIT.

VIII. More Hires Capabilities

The Apple Pascal system provides many major new capabilities to Apple II users, not the least of which is a powerful text editor and file manager. Unfortunately, in order to obtain the full benefit of this system, an external terminal has been required-until now. This disk volume was designed to overcome this limitation by providing upper and lower case text capability on an unmodified Apple II by software character generation. It is not intended to replace the existing Pascal volumes, but rather to supplement them in the same sense as a SOROC1 OR HAZEL1 disk.

All that is required to use this diskette is to insert it into drive 1 of slot 6 (Pascal device #4), and reboot the system. No changes to existing programs are required. With HIRES1, Apple Pascal users now have access to upper and lower case text for program and document editing. Also, since the Hires graphics screen is used instead of the primary text display (in Hires Mode), mixed text and graphics are automatically available under program control. Additional system and library procedures have been provided to support these features at the user level. The use of Hires upper/lower case text should cause no noticable reduction in printing speed, although screen scrolling is slightly slower.

Support has also been provided for two popular hardware modifications: the upper/lower case character generator ROM and the game I/O shift key detect wire. Both of these are already in common use and are advocated by many users' groups. This system is designed so that it will take advantage of these modifications if they have been installed, but will work just as well without them.

None of these capabilities require any additional storage other than the usual 8K for the high resolution graphics screen in Hires mode.

IX. System Configuration

When the HIRES1 diskette first boots, you should immediately notice that your Apple is displaying the Pascal "welcome" message in upper and lower case text. This is because you are looking at the high resolution graphics screen. The new BIOS (Basic I/O System) provided on this diskette is writing graphics onto this screen at very high speed to simulate an upper/lower case graphics terminal.

The file SYSTEM.CHARSET contains the block images of the system default character set, which is automatically loaded by the operating system during initialization. It resides in storage at hex location 0800 (decimal 2048) in text page 2. The storage format is identical to the Turtlegraphics character set definition and is also used by the procedures WSTRING and WCHAR.

In normal operation, the Pascal system puts programs above the Hi-Res graphics page, which reduces the space normally available for user programs by 8K. An additional option is now available at the Command level:

Mode: H(ires, L(ores, <sp> to Quit

By selecting Lores mode, you can force the system to revert to normal (upper case only) text mode and reclaim the 8K screen area. This is necessary when running certain programs on volume APPLE3:, such as the FORMATTER, and is highly recommended when doing any extensive Filer functions, such as volume transfers.

The system will automatically go to Lores mode when entering the Compiler or Assembler, so that the maximum amount of memory will be available for these functions. If you were in Hires mode before compiling, the system will return to Hires mode afterwards. If control returns immediately to the Command level, however, it will stay in Lores mode until you press a key, so that you can read the messages left on the screen before proceeding.

X. Text Editing

The editor on this disk volume is almost identical to the one contained on the volumes APPLE0: and APPLE1:. The default setting of line width to 40 characters and a reduction of file size to 10K in Hires mode are the only major changes in its configuration. Text files which extend beyond column 40 may have to be re-margined before they can be edited. The full 18K text editing capacity is available in Lores mode.

The control characters previously used for horizontal scrolling control have been implemented for upper/lower case control from an unmodified Apple II keyboard:

Control-A: Alpha lock (toggle)

Control-Z: One character upper shift (to capitalize a word)

The action of these keys is immediate and transparent to the system. The upper and lower case text you see on the screen in Hires mode is actually being received as you see it from the Apple's keyboard (what you see is what you get). In Lores mode, upper case characters will appear to be the same as lower case unless you have selected reverse video mode, in which case upper case letters will be inverted.

The action of the control-A and control-Z keys parallel the action of a typewriter's shift lock and shift keys. The control-Z key is the functional equivalent of pressing the shift key for one character, then releasing it: it conveniently leaves you in lower case, regardless of your previous mode. The control-A key toggles (alternates) shift lock mode.

If you have modified your Apple to allow the keyboard shift key to be read from the game I/O connector (switch #2), the new BIOS will automatically recognize upper and lower case text using the shift key. With this modification, Shift-M, N and P will be upper case. Pressing both the Control and Shift keys will allow access to the "]","©" and "@" symbols.

In its current configuration, the Editor uses the escape key to signify acceptance and control-C to throw away results since last acceptance. This is a potential convenience to someone who uses only this Hires system, but may be troublesome to someone who is used to the normal Pascal editor. Note that if you try to accept an insertion with control-C as you do with the Apple1 editor, you will throw away your work. You can reconfigure this system to work in the same way as APPLE1 by writing SYSTEM.MISCINFO from an Apple1 disk onto the disk you use to boot up the Pascal animation system. Control-C and the escape key will then work in the usual fashion.

XI. Programming Features

A new library unit is available to programmers using the HIRES1 system: SETSCREEN. It contains one procedure, SCREENMODE(MODE:INTEGER). The argument range is from 0 to 3. To include this procedure in your program, just add the statement: USES SETSCREEN immediately after the program name declaration.

The system normally comes up in screenmode 1 (Hires print). Lores mode is equivalent to screenmode 0. The other two modes

are (2) Hires overprint, and (3) Hires exclusive-or. Programs may switch from one Hires mode to another at any time without restriction by printing the control sequence <esc> <0..3>. It is strongly recommended, however, that you use the SCREENMODE procedure when going from Hires to Lores mode.

A special feature is provided by the SCREENMODE procedure: if screenmode zero is set before any variables are allocated on the heap and the variable space is not released, the system will reclaim the Hires screen's 8K for your use and will prevent your return to Hires mode. An example might be if you issue a NEW(P) in a program and do not issue a RELEASE before attempting to switch back to Hires. However, if you claim space on the heap before entering Lores mode the heap pointer will not be moved, and you may return to Hires at any time. This is done automatically by Turtlegraphics and other intrinsic library units at run time. An added benefit of this feature is that a complex screen can be built up on the Hires page, with mixed text and graphics or whatever, and left temporarily by switching to Lores mode. The Lores screen can then be erased and filled with text (a menu, for instance). After the user acknowledges the information with perhaps a keypress, you can return to Hires mode and your original screen will have been untouched.

XII. More Control Characters and Some Tricks

A few more control characters complete our discussion of HIRES1:

Control-E: Embedded carriage return and linefeed

Control-N: Set normal mode (white on black)

Control-R: Set reverse mode (black on white)

Control-T: Set full (40x24) text window

Control-V: Set upper-left corner of window

Control-W: Set lower-right corner of window

The proper use of these is left to your imagination. Here are a couple of experiments you may find interesting, though...

(1) Get into the Filer (F), and get a directory listing of drive 4 (#4:). Now hit (E) again when it stops listing the files. You should see the prompt: "Extended directory of ?". Now type control-V to set the upper-left corner of the text window. Don't worry- you won't see any visible change- yet. Now complete the entry by typing (#5:) and hit return (you do have a disk in the second drive, don't you?). Surprise! When you're finished comparing the two directories, just press control-T to release the window.

(2) At the command level again, press control-R to reverse the video screen. Notice how much clearer the text is on a color TV?

(3) Press (ML) to switch to Lores mode. Now switch back to normal mode with a control-N. Notice the change in the capital letters?

(4) Have fun- Experiment!

XIII. Example of Program Development

Here is an entry level program to demonstrate how to get started with the Pascal Animation Tools. This program will simply draw a ship and move it across the Hires screen. Several more advanced programming examples are provided in source code on HIRES1 and HIRES2.

Let's begin by drawing the ship. From command mode, X)ecute CHAREDIT and when prompted for the name of the character set, type SHIP. Since this character set does not yet exist, a new file will be created. Had there already been a character set by that name, it would have been loaded. If you are familiar with either the Pilot Character Editor or the DOS TOOLKIT, you will find that CHAREDIT functions virtually identically.

If you are not familiar with any of these other products, you will need the Help Screen within CHAREDIT. To enter it, type <esc> and then "?". (You did already bring up CHAREDIT and had the Editor Screen with the the three squares, didn't you?) Here is what the Help Screen looks like:

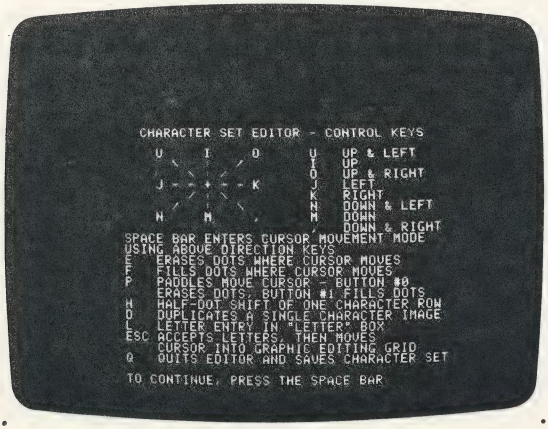


FIGURE 1. Help Screen

Follow its instructions to either copy the ship or create your own figure. Use either the game paddles or the keys indicated, according to your own personal preference. Here is what our ship looked like, when it was finished:

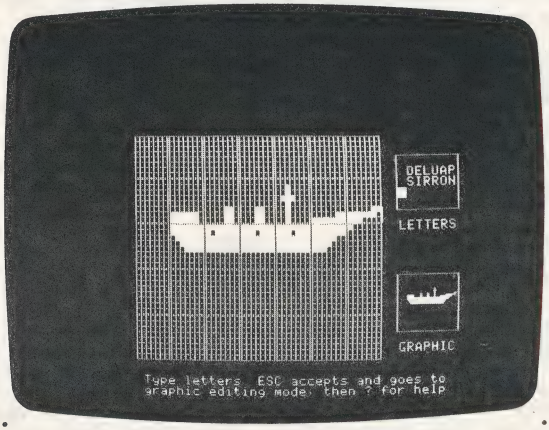


FIGURE 2. Ship

There are two main points to notice about this figure. First of all, the actual letters which are used to make up parts of the figure are of no significance. Any other letters,digits or characters could be used; the only restriction is that (obviously) the same letter or character cannot be used to represent more than one 5X7 dot pattern within the same character set. However, you may have two or more letters represent the same pattern, if you wish. The second point is simply to note how the grouping of letters in the upper right hand box corresponds to the parts of the figure in the expanded graphics box. Thus, in this character set the letter "U" corresponds to the ship's mast, and the letters "E" and "L" correspond to the two smokestacks. Note also that the letters "SIRRON" form the lower part of the ship. Thus, wherever we print these 12 characters in this pattern while using the character set "SHIP.FONT", we will have a picture of a ship on the screen. Save this character set by typing "Q" from graphics mode (not letter mode) and answering the exit question(s) appropriately.

Now we get to the text part of the program. The lines which are entirely capitalized are program lines; the rest are explanatory

in nature.

```
PROGRAM SHIPP;

USES ANIMATION,APPLESTUFF;
```

Note that ANIMATION is in the library provided on HIRES1. It is not necessary to have ANIMATION.CODE present in the system in order to run programs.

```
VAR X,Y,RIGHTEDGE:  INTEGER;
SHIP:  FONT;
FRAME0, FRAME1:  PICTURE;
THEENDOFTIME:  BOOLEAN;
```

SHIP is the character set which we defined previously.

```
PROCEDURE SHIPDRAW;
BEGIN
BLOCK(FRAME0,' DELUAP / SIRRON ',RIGHT);
BLOCK(FRAME1,'ZZZZZZ /ZZZZZZ',STAY)
END;
```

There are several things to note about this procedure. First of all, FRAME1 is used to erase the ship when it reaches the end of the screen, so "Z" is a character which has been given no graphic definition and will print as a blank. Spaces could also be used; note that they have been used behind the stern of the ship in order to erase the images which have been left behind. The procedure ANIMATE, which is called later, automatically maintains any images it draws until they are explicitly erased. Note that what procedure BLOCK actually accomplishes is to define FRAME0 and FRAME1 and tell how to move the cursor after drawing each of them. Each of the lines beginning with "BLOCK" is actually acting in a manner analogous to an assignment statement. Look at the second argument of BLOCK, which is contained within the quotes. This is the configuration of characters which was earlier defined in order to give the appropriate screen image. You may want to look back in order to see the screen where we defined these characters. Note that the slash signifies to break to the next row. Note also the movement command at the end of each block procedure; this tells how to move the cursor at the end of drawing each frame. In this case, the movement of the ship is accomplished by moving the cursor to the right after each execution of ANIMATE.

```
BEGIN
RIGHTEDGE:=38;
LOADFONT(SHIP,'SHIP.FONT');
SHIPDRAW;
USEFONT(SHIP);
THEENDOFTIME:=FALSE;
REPEAT
   X:=0;Y:=5;
   REPEAT
      X:=0;Y:=Y+1;
      REPEAT
         ANIMATE(FRAME0,X,Y,PADDLE(0));
         IF KEYPRESS THEN
            BEGIN
            STDFONT;
            EXIT(SHIPP)
            END;
      UNTIL X>RIGHTEDGE;
      ANIMATE(FRAME1,X,Y,PADDLE(0));
   UNTIL Y>20;
UNTIL THEENDOFTIME
END.
```

At this point the program has used each of the five procedures included in the ANIMATION library unit. BLOCK was used previously in order to define the two frames we are using. LOADFONT is used at the beginning of the main program in order to load our graphic character set into memory. USEFONT is called in order to tell the program to actually begin doing all printing to the screen in this character set. Then ANIMATE is called to place and maintain the two frames on the screen. Note that ANIMATE automatically updates X and Y (the second and third parameters fed into it) to reflect the movement of the cursor when the frame is finished. The fourth parameter fed to the ANIMATE procedure is a pause before going on to the next program step. In this program, PADDLE(0) controls how fast the ship moves.

For instructional purposes, this program was kept at minimal complexity. It consists of an unchanging figure which merely moves across the screen. The movement is not smooth, because the ship is moving a whole character square at a time. Using more characters and moving the ship by less than a whole square would result in much smoother movement. For a more complex example, list HORSE.TEXT which actually animates a figure by switching between different images. Slightly more complex yet is TROT.TEXT which animates the figure and also moves it across the screen.

APPENDIX A

SETTING UP THE APPLE II SYSTEM

This appendix includes a list of the equipment you'll need to use the programs on your Apple II. You do not need to read all the manuals, but they should be on hand to answer questions that may arise in operating the equipment (e.g., how to boot a diskette).

To use the Pascal Animation Package, you'll need the following equipment:

- an Apple II or an Apple II Plus with 48K bytes RAM and the Apple Language System;
- an Apple Disk II with Controller (16-Sector PROMs);
- a Video Monitor or Television.

For reference, you should have on hand a copy of the following manuals:

- This Manual (A User's Guide to the Programs);
- an Apple II BASIC Programming Manual (Setting up the Apple II);
- an Apple Pascal Reference Manual.

Putting The Pieces Together

Here are the steps to follow to put your system together:

(1) To set up your Apple II, follow the instructions in the Apple II BASIC Programming Manual. You may not need to attach the Game Controllers, although there is no harm in doing so. Your Apple II must have at least the minimum amount of memory listed under the equipment description for you to use the programs.

(2) If you already have a Disk Operating System, and are using a version of DOS that runs in 13 sectors (DOS 3.2.1 or earlier), you will need to change two proms on your disk controller card to update your system to 16 sectors. Any version of DOS earlier than release 3.3 will need to be updated. These proms are also the same proms that come with the Pascal Language System. Consult a DOS 3.3 manual for these procedures.

APPENDIX B

Notes Regarding Copy Protection

Special Delivery Software is copy protected, except in the case of program utilities or template-applications for major products (e.g. Apple PILOT).

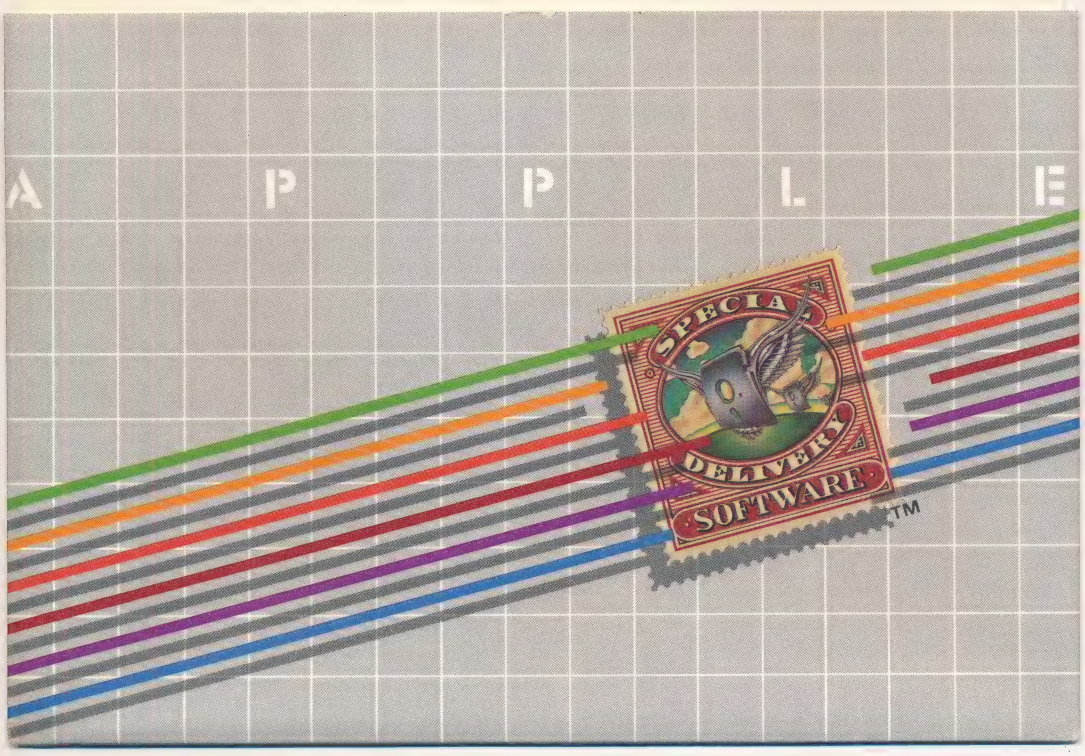
In order to provide you with a backup capability, we enclose duplicates of your diskettes. You should definitely store your backups in a safe location, and NOT use them. In the event a protected diskette becomes damaged within the time period of Special Delivery Software's media warranty, you may return the diskette to us for replacement, and continue to use your backup until we can send you a replacement.

Unlike most other copy protection schemes, our method is selective; protected and unprotected files may reside on the same disk. In the PASCAL ANIMATION TOOLS, CHAREDIT.CODE is the only protected file. Any other files may be copied freely onto backup diskettes. Also, to use this software you must realize that the BIOS in this package is incompatible with normal Apple Pascal. Therefore, you will find that programs which utilize these animation procedures must be run after booting from HIRES1. However, programs which you write within this system can be compiled and linked using SYSTEM.COMPILER and SYSTEM.LINKER from Apple Pascal 1.0.

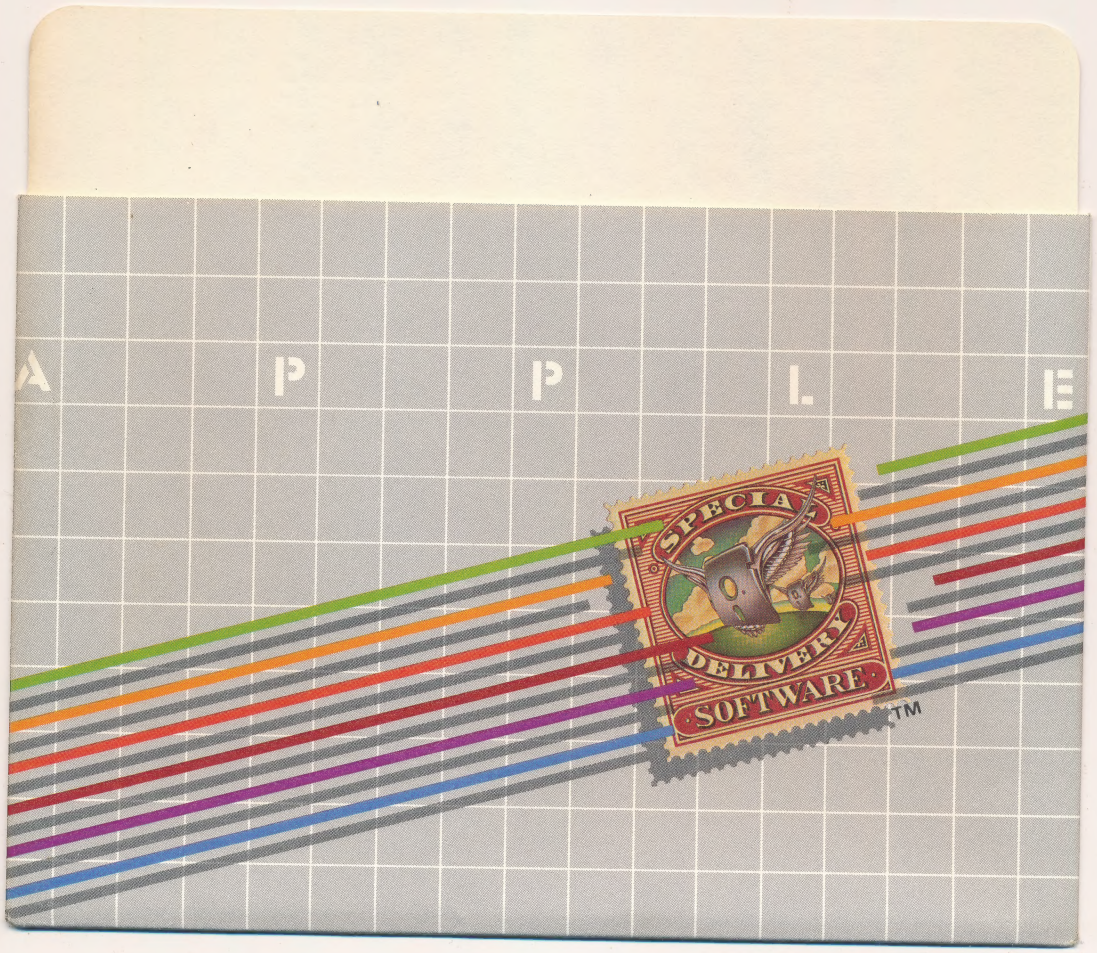
To conserve your use of the original diskettes you may wish to adopt a scheme similar to the following. Format diskettes using your normal Pascal system. Then boot from HIRES1 and use its Filer to transfer all of its files beginning with SYSTEM to one of your formatted diskettes. You may also transfer any of the other files you are interested in except CHAREDIT.CODE. Attempting to transfer CHAREDIT or the entire disk will result in an I/O error. No harm will be done, but the transfer will be aborted. If you want the Editor in this system to use control-C and <esc> in the same fashion as the normal Pascal editor, then transfer SYSTEM.MISCINFO from APPLE1 onto the HIRES disk that you are creating. If you have a second drive you can use APPLE2 while developing programs in this system. The new HIRES disk that you have created can boot and be used to develop animation programs. The only exception is that when you develop your new character sets, you must load CHAREDIT from one of the diskettes provided with the PASCAL ANIMATION TOOLS.

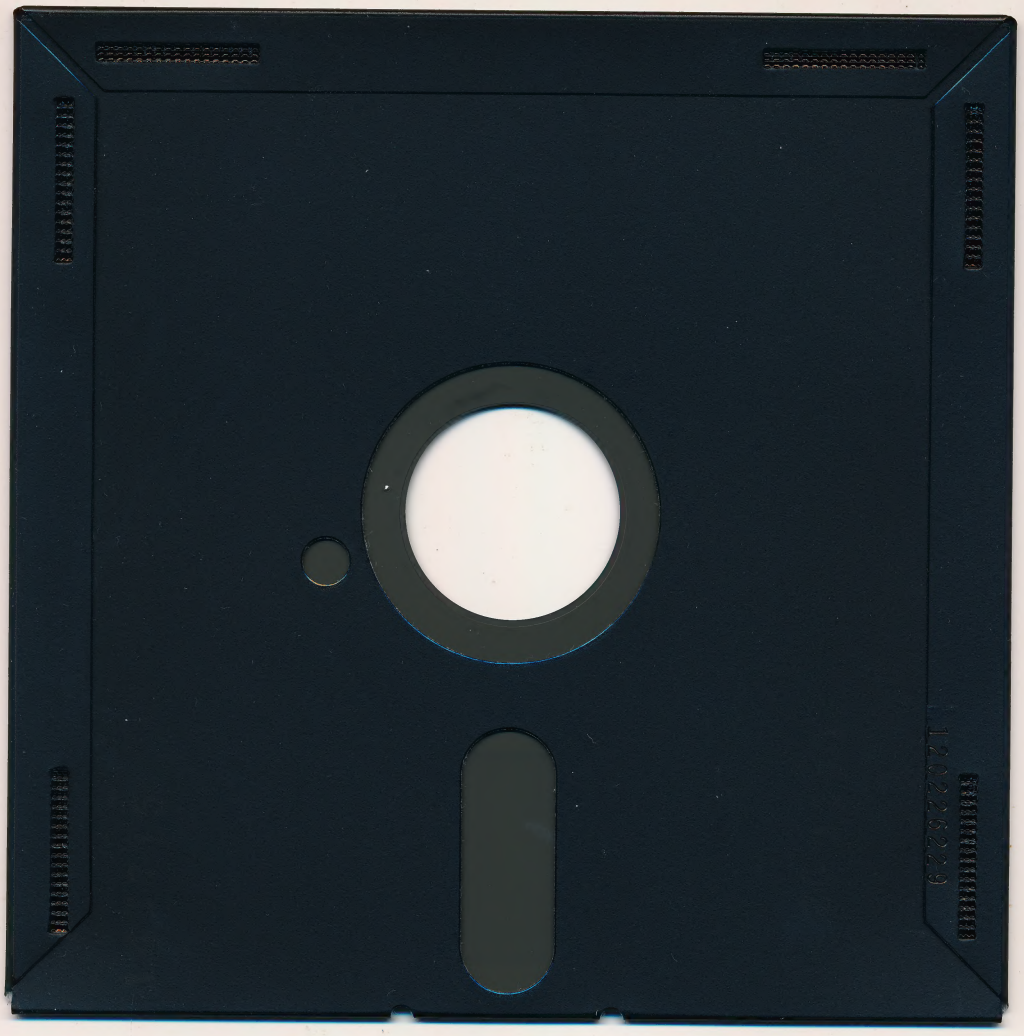
One final and very important point: do not use Apple Pascal 1.1 in conjunction with this package in any way. They are totally incompatible. Do not even use the Formatter program from 1.1 to initialize diskettes to use with this system.

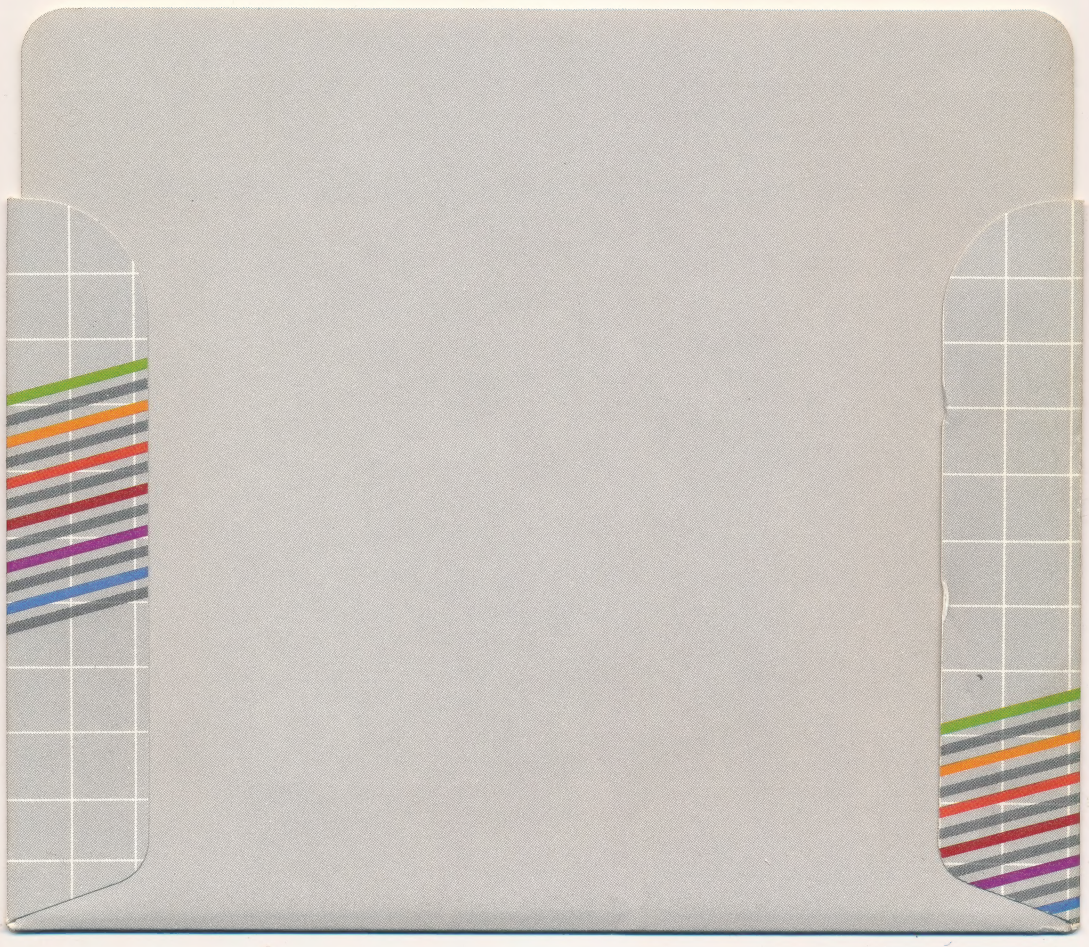

SPECIAL DELIVERY SOFTWARE
PASCAL
ANIMATION TOOLS
VOLUME 2
C2B0001
©1981 Apple Computer Inc.
652-2020-01


APPLE
SPECIAL DELIVERY SOFTWARE


SPECIAL DELIVERY SOFTWARE
PASCAL
ANIMATION TOOLS
VOLUME 1
C2B0001
©1981 Apple Computer Inc.
652-2019-01


A P P L E
SPECIAL DELIVERY SOFTWARE





SPECIAL DELIVERY SOFTWARE™

CUSTOMER LICENSE AGREEMENT

Dear Customer:

IMPORTANT: The SPECIAL DELIVERY SOFTWARE™ Product that you have just received from APPLE is provided to you subject to the Terms and Conditions of this Software Customer License Agreement. Should you decide that you cannot accept these Terms and Conditions, then you must return your product with all documentation and this License marked "REFUSED" within the 7 day examination period following receipt of the product.

1. License. APPLE hereby grants to you upon your receipt of this product, a nonexclusive license to use the enclosed SPECIAL DELIVERY SOFTWARE™ product subject to the terms and restrictions set forth in this License Agreement.

2. Copyright. SPECIAL DELIVERY SOFTWARE™, including its documentation, is copyrighted by APPLE or, in some cases, by APPLE's software suppliers. You may not copy or otherwise reproduce the SPECIAL DELIVERY SOFTWARE™ or any part of it except as expressly permitted in this License. Any SPECIAL DELIVERY SOFTWARE™ product that is not copy protected may be copied for backup use only, provided that you reproduce all copyright notices and other proprietary legends on such copies.

3. Restrictions on Use and Transfer. The original and any back-up copies of SPECIAL DELIVERY SOFTWARE™ are intended for your personal use in connection with a single computer. You may not distribute copies of, or any part of SPECIAL DELIVERY SOFTWARE™ to others without the specific granting of a Software Distribution License from APPLE for that purpose.

4. Limited Warranty on Media. APPLE warrants the diskettes on which SPECIAL DELIVERY SOFTWARE™ is recorded to be free from defects in materials and faulty workmanship under normal use for a period of 90 days after the date of purchase. If during this 90-day period, a defect in the disk should occur, the diskette and a copy of your receipt may be returned to the SPECIAL DELIVERY SOFTWARE™ Operation at APPLE, and APPLE will replace the diskette without charge. Your sole remedy in the event of a defect in the diskette is limited to the replacement of the diskette as provided above.

5. LIMITATIONS ON WARRANTY AND LIABILITY. EXCEPT AS EXPRESSLY PROVIDED FOR MEDIA, APPLE, ITS SOFTWARE SUPPLIER, DISTRIBUTORS AND DEALERS MAKE NO WARRANTIES, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THE SPECIAL DELIVERY SOFTWARE™, ITS MERCHANTABILITY OR ITS FITNESS FOR ANY PURPOSE. SPECIAL DELIVERY SOFTWARE™ IS LICENSED SOLELY ON AN "AS IS" BASIS. THE ENTIRE RISK AS TO ITS QUALITY AND PERFORMANCE IS WITH YOU. SHOULD THE SPECIAL DELIVERY SOFTWARE™ PROVE DEFECTIVE, YOU (AND NOT APPLE, ITS SUPPLIER, DISTRIBUTOR, OR DEALER) ASSUME THE ENTIRE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION AND ANY INCIDENTAL OR CONSEQUENTIAL DAMAGES. IN NO EVENT WILL APPLE, ITS SUPPLIER, DISTRIBUTOR, OR DEALER BE LIABLE FOR DIRECT, INDIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT IN THE SOFTWARE, EVEN IF THEY HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF IMPLIED WARRANTIES OR LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE LIMITATIONS MAY NOT APPLY TO YOU.

SPECIAL DELIVERY SOFTWARE™
10260 Bandley Drive
Cupertino, CA 95014