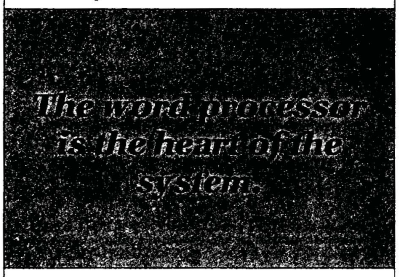
The SwyftCard: Jef Raskin's New User Interface

by Dave Caulkins



SwyftCard
Information Appliance Inc., 1014 Hamilton Ct., Menlo Park, CA 94025, (415) 328-5160; $89.95

The SwyftCard from Information Appliance is a new personal computer environment for the Apple IIe and IIc. It provides a word processor from which you can easily perform other tasks, such as disk file access, calculations via BASIC, printing, and communications with a modem. The environment is noteworthy because it is philosophically at odds with the popular interface made of icons, windows, and mouse that is used on the Macintosh and other computer systems. Significantly, SwyftCard's designer is Jef Raskin, one of the initiators of the Macintosh project at Apple Computer.

I tested SwyftCard on an Apple IIe, but as this issue goes to press, Information Appliance has announced a version for the IIc also. To use SwyftCard on a IIe, you need an 80-column card, a video monitor suitable for 80-column display, and a disk drive. The SwyftCard supports a wide variety of printers; several require no setup, and many can be installed by using the Calc command.

SwyftCard consists of a three-chip board that plugs into slot 3, a set of stick-on labels for several keys, a tutorial disk, and an excellent manual. The tutorial will help you get started with SwyftCard by guiding you through a series of short lessons on the use of its features.

The word processor is the heart of the SwyftCard system. It has been carefully designed to achieve several goals:

- Speed: For text processing and floppy-disk access, SwyftCard is significantly faster than more expensive systems. All the text is in RAM, so disk access time doesn't slow SwyftCard down. Half the system is implemented in tokenized Forth and half in assembly language, resulting in fast operation. Each SwyftCard file occupies a single floppy disk and can expand to 40,000 characters, or roughly 14 pages of single-spaced text. Disk operations take less than seven seconds in all cases.
- Simplicity: There are only ten basic commands: Insert, Delete, Print, Leap, Creep, Page, Calc, Print, Send, and Disk. The last three are for input/output and are not used as frequently as the others. Most commands are implemented with one or two keystrokes using a few dual-purpose, specially labeled, SwyftCard function keys. After a little practice, users learn the commands, and text processing becomes fast and easy.
- Optional environment: You can load an ordinary Apple disk, and the operating system works normally, as SwyftCard hides behind the scenes until you need it.

Dave Caulkins, 437 Mundel Way, Los Altos, CA 94022

How SwyftCard Works

Leaping

SwyftCard allows you to move through text using two Leap keys: the open-apple and closed-apple keys on either side of the space bar. To make learning easy, a set of stick-on labels is provided to indicate which keys are used for special SwyftCard functions. Leaping means moving from one place in the text to another immediately.

Suppose the phrase *The number is less than the numerator* appears between the cursor and the end of the file and you want to locate the cursor on the *n* in *numerator*. Press the right Leap key as you type "n−u−m−e" to create a search pattern. After you have typed the "n," the cursor leaps to the next instance of *n* in the file, after the *u*, to the next instance of *nu*, and so on until it stops on the *n* in the word *numerator*. (incremental search)

If this isn't the instance of the word you want, press the Leap key, and the Leap Again key (Tab), to find the next occurrence of the letters. Leap Again auto-repeats; if you hold it down for more than half a second, the cursor will move rapidly from one instance of the pattern to the next. If the cursor arrives at the end of the file, it wraps back to the beginning. It will continue its forward direction until the Leap keys are released.

If you wish to leap to a point between the cursor and the beginning of the file, use the same procedure

with the left Leap key, and the search will take place in the reverse direction.

Lowercase letters in a leap pattern match both uppercase and lowercase letters in the text. Uppercase letters in the pattern match only uppercase in the text. You can leap from word to word (press Leap and the space bar), paragraph to paragraph (press Leap and Return), or page to page (press Leap and the Page key [Esc]).

Creeping

Moving the cursor over just a few characters is called creeping in the SwyftCard manual. To creep, press and release the right Leap key, and the cursor moves a character to the right. To creep in reverse, press the left Leap key.

Deleting

Deletes can also be done to the left or right of the cursor. While you are typing, if you press the Del key, the character to the left is erased as with the backspace key on a typewriter. After you leap or creep to a new place in the file, characters to the right of the cursor are deleted. The appearance of the cursor itself indicates which delete is operative. A narrow cursor is one character wide and appears when right delete is in effect; the wide cursor is two characters across with the cursor and the reverse-video character split.

Highlighting

Another useful SwyftCard feature is the highlighting of text. Highlighted text can be deleted, saved in a buffer for later insertion, printed, or telecommunicated. Highlighting takes place by pressing and releasing a Leap key, moving the cursor, and then pressing both Leap keys simultaneously. Any amount of text can be highlighted, from two characters to the entire document. Pressing Delete will remove all highlighted text from the screen and place it in a buffer. To restore the text, locate the cursor where you want the text and press Insert (Control-A). The deleted text remains in the buffer until you highlight and delete other text.

The Disk Command

You use the Disk key to read from or write a file to the disk. SwyftCard's method of handling disk files automatically takes care of whether reading or writing is required. Let's say you want to save some text. Put a floppy disk in the drive, and press the Disk key. SwyftCard notes if the disk is empty and that text is in RAM and deduces that a disk write is needed. When text has been saved, the cursor blinks rapidly to indicate that RAM and disk contents are identical.

On the other hand, if you are in the middle of writing and try to load a new file from a different floppy, SwyftCard will observe that some changes in RAM have not been saved and refuse to overwrite. A beep serves to remind you that the disk for the old file should be inserted in the drive to save changes. If you really do not want to save your changes, the procedure is to delete the text. These are examples of a well-planned user interface, designed to save the user from accidental, catastrophic errors, which all computer users have experienced at some time. Another example of this care is a SwyftCard com-

SWYFTCARD
(continued from page 43)

mand that will make a disk look blank, in effect destroying anything on it and allowing you to write to it. Because this command can affect your data and is thus dangerous, it does not use the Disk key and is difficult to execute accidentally.

The Calc Command

The Calc command causes a BASIC statement to be executed in the file; for example, if you type "?34 + 78" and then highlight it and press the Calc key, the answer 112 will appear in place of the BASIC statement in your text. More complex executions are also possible. If you type

```
10 FOR I = 1 TO 31
20 PRINT "JANUARY"; I
30 NEXT I
RUN
```

then highlight this program and press the Calc key, the following calendar for January will be placed in your file where the program was:

```
JANUARY 1
JANUARY 2
JANUARY 3
.
.
JANUARY 30
JANUARY 31
```

Almost all BASIC commands will work, but the size of the BASIC program is limited to 900 bytes in the compacted internal form. Some uses of BASIC are dangerous—*CALL*, *PEEK*, and *POKE* can zap your file if used incorrectly or with values that interfere with the SwyftCard.

Awkward Moments

When the SwyftCard has used all available RAM memory, it will beep each time you press a key. To create room, execute at least two deletions. One is insufficient because the delete buffer continues to take up the same amount of RAM. At this point it is clearly advisable to save the file, insert a new disk, delete part of the text, and continue. This is one of the few awkward operations of SwyftCard. An improvement would be some sort of warning when all but 50 or 100 bytes of RAM had been used to allow for a more graceful conclusion of an editing session.

The SwyftCard approach of allowing each disk to hold one file is acceptable for a machine such as the Apple II, but as memory and hard disks continue to drop in price, the SwyftCard environment will have to adapt to machines with greater internal and external storage capabilities. These implementations will require a file management system and some scheme for mapping files into a range of RAM memory sizes.

These minor complaints are far less important than the many impressive characteristics of SwyftCard, including speed, ease of use, and diversity. Overall, SwyftCard offers a strikingly innovative user interface that deserves the attention of users and software developers interested in advancing the cause of usable computers.

DDJ

Vote for your favorite feature/article.
Circle Reader Service **No. 5.**