Chapter 1

Computers, Programming, and Windows

1. Visual Basic as a Tool
2. Using a Computer
3. The Basics of Working in Windows
4. Writing Programs

GOALS

After working through this chapter, you will:

- Understand how you can use Visual Basic to solve a wide range of problems.
- Be able to identify the hardware components that make up a desktop computer.
- Learn how the computer loads and runs programs.
- Be able to explain how programming has changed since the 1950s.
- Understand the difference between procedural and event-driven programming.
- Use Windows commands and applications to manipulate text and files.

OVERVIEW

You will find Visual Basic to be an excellent tool for solving a variety of problems, such as organizing your CD collection or estimating the number of tadpoles in a pond. You can also use Visual Basic to play with graphics. For example, you could draw a picture freehand, then use Visual Basic to change the color of the drawing from red to green every 60 seconds.

This chapter provides some of the background you may need to review in order to explore Visual Basic. An introduction is provided to computer hardware, and the components of a computer are defined. You will also learn how a computer runs programs by storing, loading, and executing files.

With Visual Basic, you are a programmer. You will create programs that you or your friends can run. This task of programming is much easier with Visual Basic than it used to be. Even one or two decades ago, programmers had to write much more code to run a program than you do now. This chapter describes some of these changes in programming methods and languages.

The programs you will create with Visual Basic are computer files. Before you create a program, you need to know how to create, save, and delete files using Windows. This chapter covers these procedures, as well as how to organize files into directories. Finally, you will have the opportunity to become comfortable with the Windows Program Manager, File Manager, and Notepad.

Visual Basic as a Tool

Professional programmers use Visual Basic every day to meet a range of needs in business, industry, and science. You will use Visual Basic to perform the same kinds of tasks that these programmers tackle. For example, you can use this tool as a simple text editor, as shown in Figure 1-1. The vertical bar at the end of the passage shows that the text can be edited.

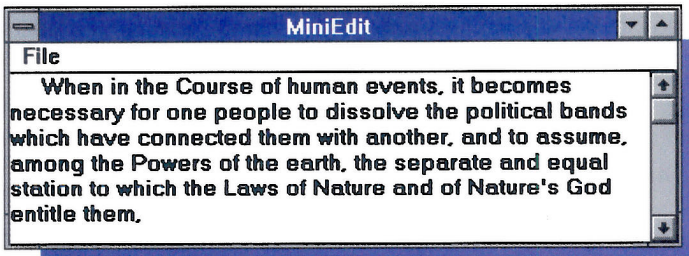


Figure 1-1
Visual Basic used as a text editor

You can also write programs that read and write files. The program shown in Figure 1-2 reads two files: the first contains a class list; the second, a list of athletes. Both lists are displayed along with a list of students who are both in the class and athletes.

Yet another way to use Visual Basic is to solve mathematical problems. The program shown in Figure 1-3 uses the coordinates of two

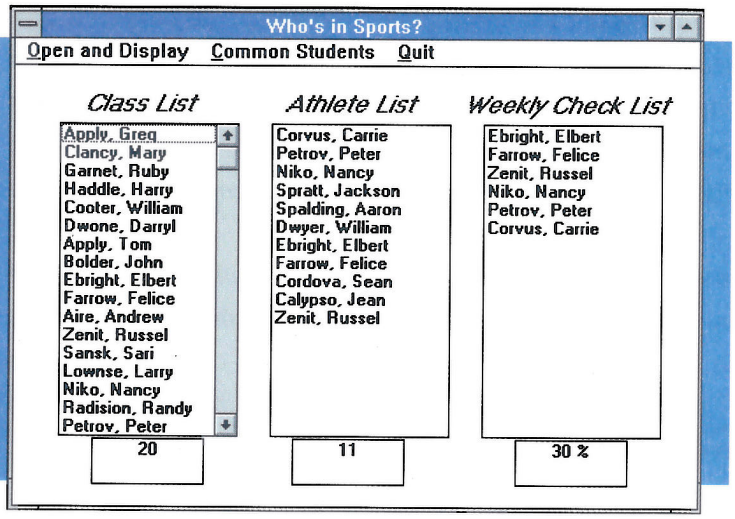


Figure 1-2
Reading two files

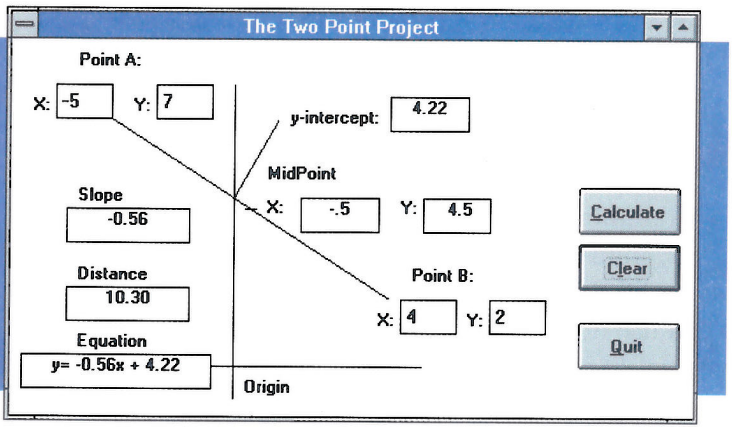


Figure 1-3
Calculating information about the connecting two points

points to calculate information about the line connecting the points. Visual Basic fully automates the calculations.

You can write programs for drawing graphics. You can set up this kind of program so that the user can draw freehand or create standard shapes, such as circles and squares. See Figure 1-4.

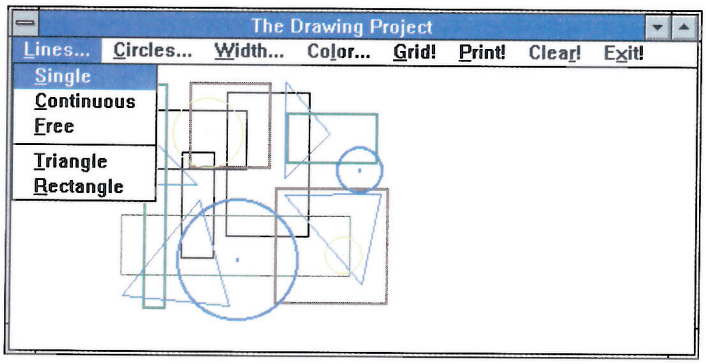


Figure 1-4
Writing a graphics program

With Visual Basic, you can take advantage of tried-and-true programs such as Microsoft Excel and Word. Why write and debug all the code you would need for a word processing program? Instead, you can open and use Word from within any Visual Basic program. Millions of person-hours have been devoted to creating Microsoft Word. You are unlikely to devote that much time to the task yourself!

These are just some of the challenges and problems you can use Visual Basic to solve. When you are finished with this course, you will be able to write applications for home, school, or any of your outside interests.

2 Section

Using a Computer

This section introduces the components of a personal computer and outlines their relationship to each other. If you are not already familiar with such concepts as RAM, CPU, and loading a file, you should read this section.

Hardware Components

The physical components of a computer are called hardware. You are used to thinking of a computer as a monitor, a box, and a keyboard. In the paragraphs in this section, you will learn what's inside the box (Figure 1-5).

CPU AND MEMORY

You can think of the central processing unit, or CPU, as the engine of the computer. The CPU does most of the work performed by the computer. It both interprets instructions contained in programs and manipulates data. The CPU can read data from two kinds of computer memory:

- RAM, or random access memory
- ROM, or read-only memory

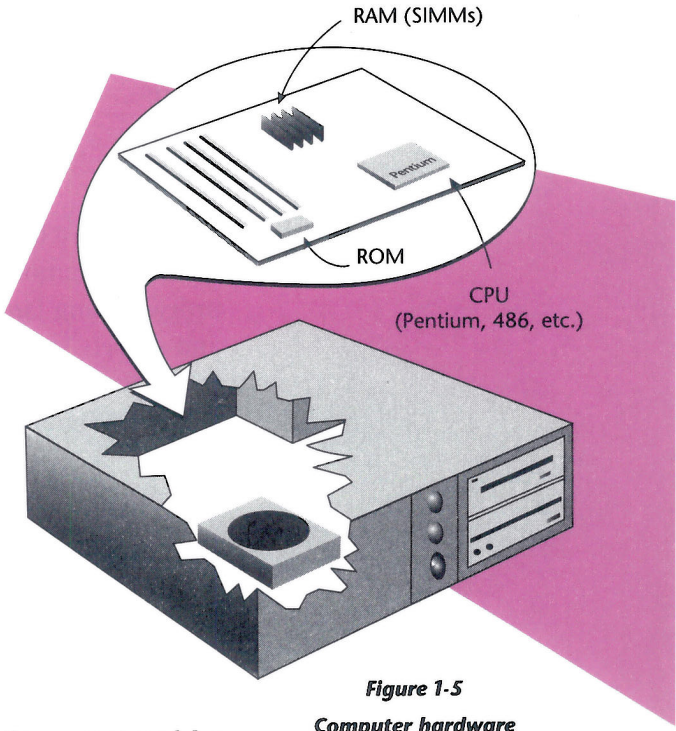


Figure 1-5
Computer hardware

RAM is used to store programs and the information processed by programs on a temporary basis. The CPU can fetch information from any area in RAM, which is why this type of memory is called "random access." When you work in a program such as Word, you are changing

information stored in RAM. If your computer loses power suddenly—for example, if you pulled the power cable out of the socket—you would lose the information stored in RAM.

In contrast, ROM is typically used only once in a computer session. The CPU reads the programs and information in ROM when you are booting up the computer, which is the term used for starting the computer. You cannot alter any of the information in ROM; it is permanent.

HOW RAM IS ORGANIZED

RAM is divided into "cells" called bytes. Each byte itself consists of eight "bits." A bit, or binary digit, is a single digit, 0 or 1. The binary number system expresses numbers with just these two digits, 0 and 1. Small numbers fit in a single byte. Large numbers take several. In the following table, you can see how the decimal system converts to binary.

In Decimal	*In Binary*
1	1
2	10
3	11
4	100
10	1010
500	1 11110100
1000	11 11101000

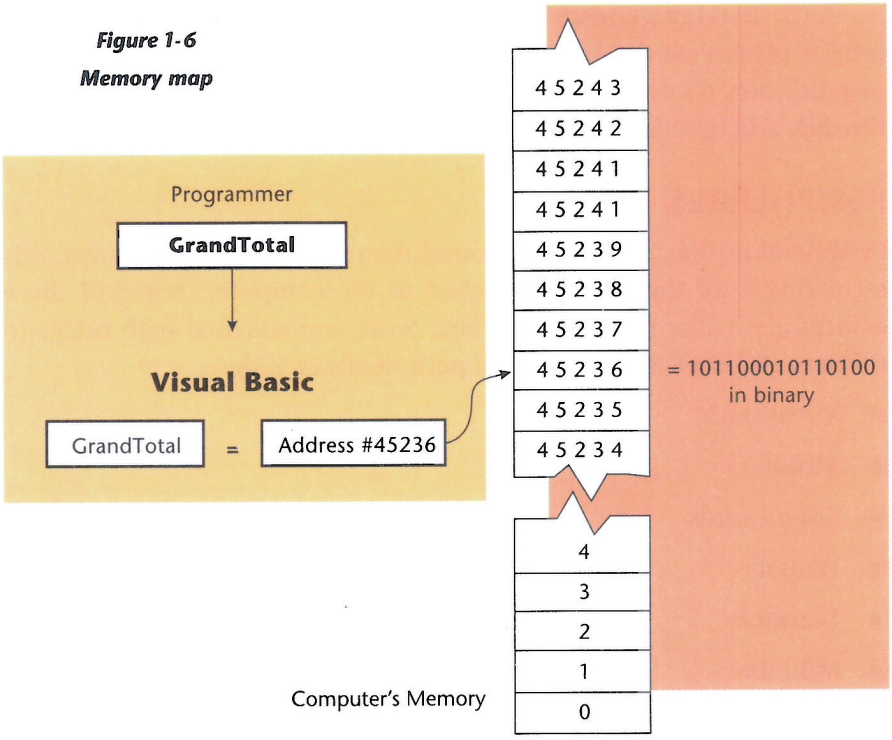
Just like a house, each cell of memory has a unique address. The CPU identifies memory cells by a numerical address (Figure 1-6). "High-level" programming languages such as Visual Basic let you refer to quantities and objects stored in memory by using meaningful names rather than numerical addresses. GrandTotal, PhoneNumberAtWork, MainWindow, and UserWantsToExit are typical of the names used.

EXTERNAL STORAGE

You keep programs and data for the computer in external storage. RAM is only a temporary storage place, and it is not large enough to hold all the programs you want to run. A personal computer can have three kinds of external storage:

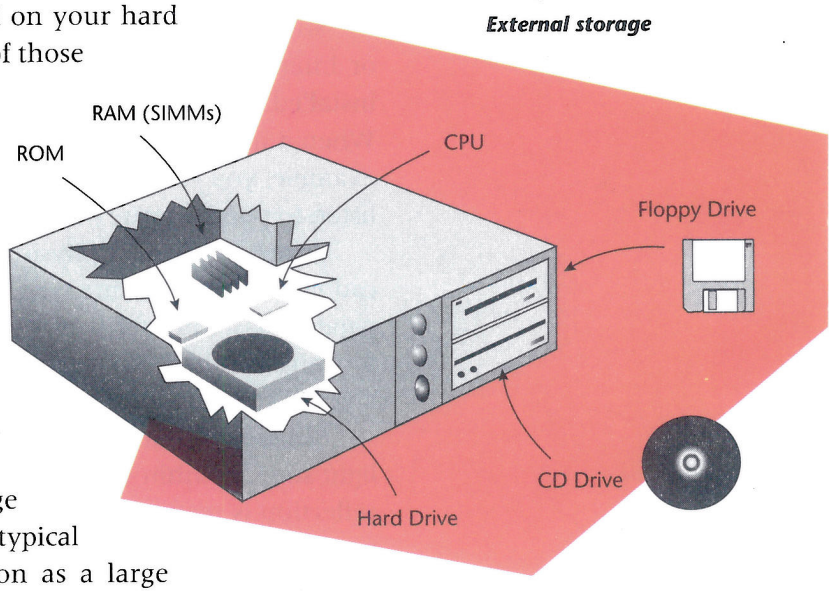
- Hard disk
- Floppy disk
- CD-ROM disk

Figure 1-6
Memory map



A hard disk is typically inside the computer (but outside the CPU) (Figure 1-7). You can store 50 to 100 times more information on a hard disk than on a floppy. For example, you probably have seven or more programs such as Microsoft Word stored on your hard drive. You probably cannot fit even one of those programs on a floppy disk.

Figure 1-7
External storage



Floppy disks and CD-ROM disks are easily portable. You can put a floppy disk in a shirt pocket, a CD-ROM disk in your backpack. Depending on the age of your computer, it may not have a drive for CD-ROM disks. Physically, a CD-ROM disk is identical to the CD disks you play in a compact disk player. This newer technology is an excellent way to transport large amounts of information at low cost. A typical CD-ROM can hold as much information as a large encyclopedia. This type of external storage is perfect for holding graphics, music, and animation files.

A CD-ROM drive reads a CD-ROM disk with a laser beam. The laser reflects off pits on the surface of the disk. The presence and absence of pits indicates 0's and 1's. The bitstream is the stream of data read from the disk and transferred to RAM.

PERIPHERALS

Peripheral means "something around the outside." For computers, this term covers all the devices attached to the computer. Some of these devices are in the main case, or box. Some are attached with cables to this box (Figure 1-8). Examples of peripherals include:

- Video cards
- Monitors
- Sound cards
- Printers
- Scanners
- Modems

Video cards are devices that translate computer output so that it can be displayed on a color monitor. How good that display is depends on the quality of the video card and monitor. With more expensive video cards and monitors, for example, you can see more colors on the monitor and images are crisper, with more detail.

Sound cards play a similar role with stereo speakers. They translate, or interface, between the CPU and the speakers. With a sound card installed, you can enjoy an added dimension of programs such as games. When a monster roars or an automobile screeches around a curve, for example, you can hear them. If you have a CD-ROM drive, you can also listen to your favorite music.

Printers provide a physical copy of the output of a computer application. An ink jet printer works by forcing a fine stream of ink onto the paper. A laser printer works by using a laser and a photosensitive drum to transfer toner to paper. The toner is a fine powder that is heated to bond to the paper.

Scanners create machine-readable images of printed material. A light source within the scanner reflects off the surface of the page. The reflection is read as a black-and-white or color image. The reflection is turned into bits and recorded in RAM. These images are stored on disks and can be manipulated by application programs. Fax machines have built-in scanners. The image of the document is transmitted over phone lines to other machines, which then reassemble the image.

Modems give computers the ability to communicate across telephone lines. You can call up another computer, for example, and transfer entire files. If you have the right software, you can send a fax rather than a file. You can also connect to the Internet, which is an international computer network. One of the many ways of using the Internet is to enjoy an online chat with someone. You type the words you want to say instead of speaking them, then they respond.

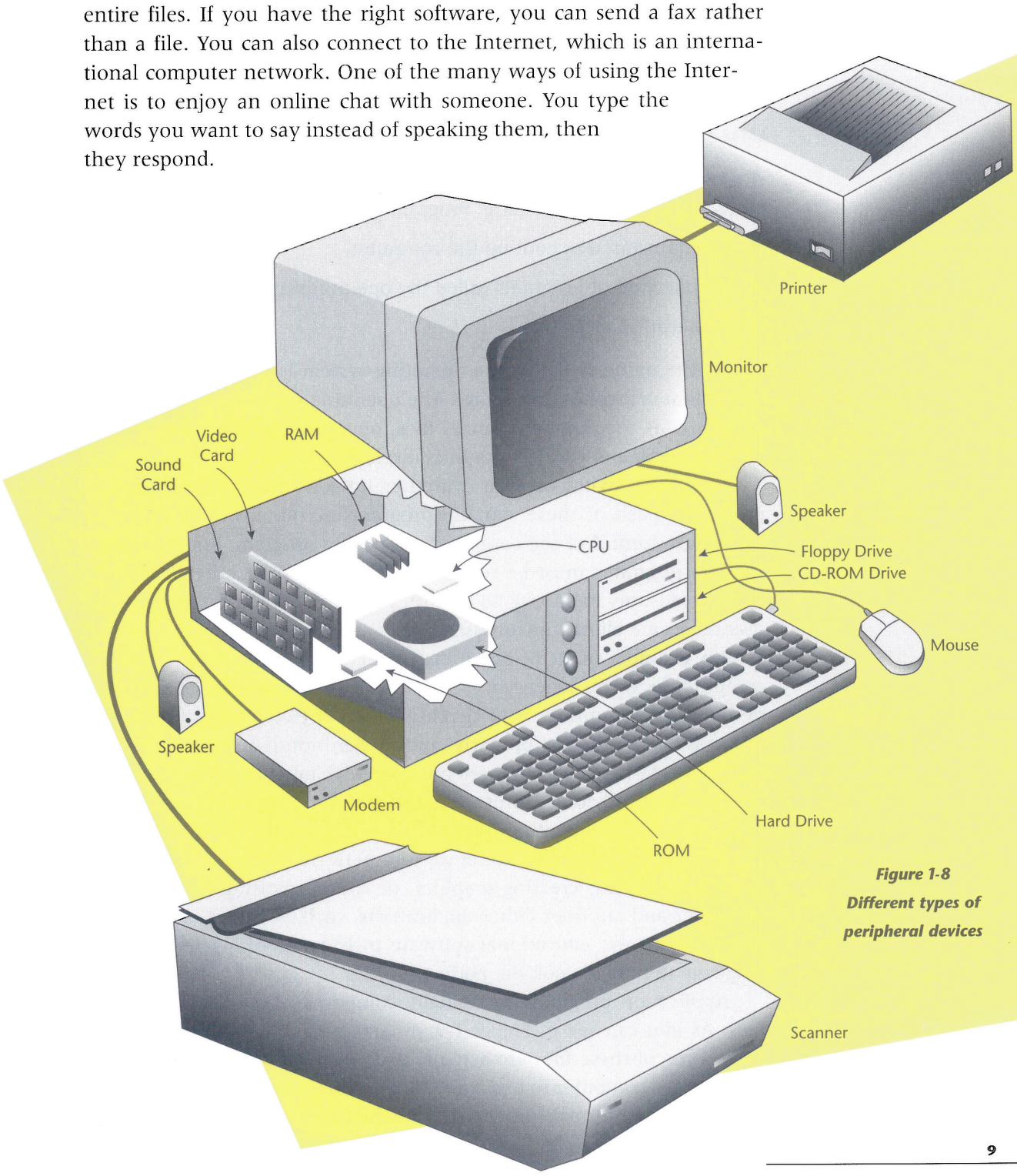


Figure 1-8
Different types of peripheral devices

How a Computer Runs Software

This section introduces computer software. Software includes the programs, both built-in and on disk, that run the computer and turn the personal computer into an information source. Categories of software and the process by which a computer runs a program are discussed.

WHAT IS A PROGRAM?

A program is a sequence of instructions to the computer in a language the computer can understand. Programs fall in three categories:

- Programs used to boot up the computer
- A collection of programs called the operating system
- Application programs

As the name suggests, the operating system is a collection of programs that control the computer. The operating system interprets commands, runs programs, organizes files, operates the disk drives, reads the keyboard, and controls graphics displays.

When you switch on your computer, it boots up by running various programs. Each of these start-up programs accomplishes its task and turns over control of the computer to another program. Some of these start-up programs must be read into RAM from disk; others reside in ROM (read-only memory), so that they can run even if the computer's disk drives are all malfunctioning.

An example of ROM-based start-up code is the self-test that the computer performs when you turn it on. An important task performed by the self-test routines is to check whether the computer's RAM and disk drives are working properly, and to alert you if any appear not to be. The ROM-based routines are not specific to any particular operating system. They obviously must run successfully before any attempt is made to read the operating system's start-up programs from disk.

Application programs are created to perform particular tasks, such as processing text, creating graphics, or communicating between one computer and another. Other applications, such as games, are just for fun. You can use educational software to learn anything from a new language to how to cook the perfect soufflé.

Application programs are usually composed of a large number of files. As you can see in Figure 1-9, programs have many components. Only some of these files are actually programs, which contain instructions for the computer to execute. The other files contain data, and they are usually in a format understood only by the application program.

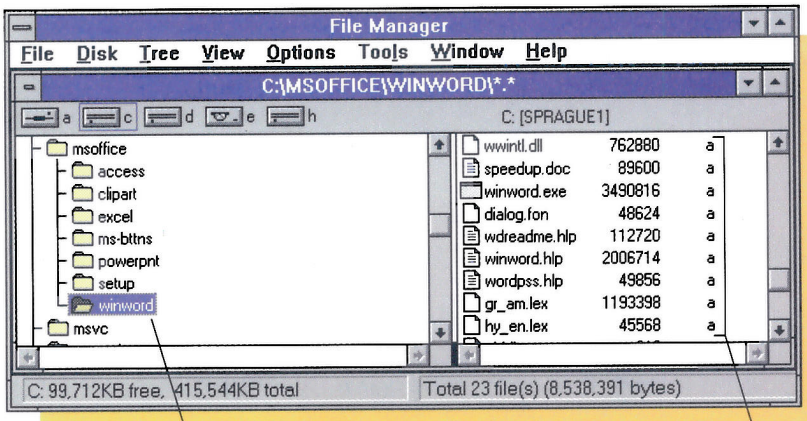


Figure 1-9
File components of Microsoft Word

For example, a word processing program such as Microsoft Word consists of:

- The executable program **winword.exe**
- Additional program files with the extension **.dll**, containing more instructions that the computer can perform
- Help files, with the extension **.hlp**, containing online help files
- Files with the extension **.lex** that help Word spellcheck and hyphenate your documents
- Style sheets and templates for form letters

LOADING AN APPLICATION PROGRAM

Application programs are kept in external storage. When you or your school buys a program, the files are provided on floppy disk or CD-ROM. Often, you will then transfer the application to your hard drive.

The operating system is responsible for loading a program from external storage (floppy disk, hard disk, or CD-ROM) into RAM. When you choose a program to open, you are sending a message to the operating system to load that program (Figure 1-10). The CPU runs applications from RAM, not from any of the external storage devices.

EXECUTING THE PROGRAM

Once the program's instructions are copied into RAM, the operating system informs the CPU of the location of the program in memory. The CPU fetches each instruction from memory, decodes it, and executes it. While the program runs, it uses RAM to store the data it manipulates, as

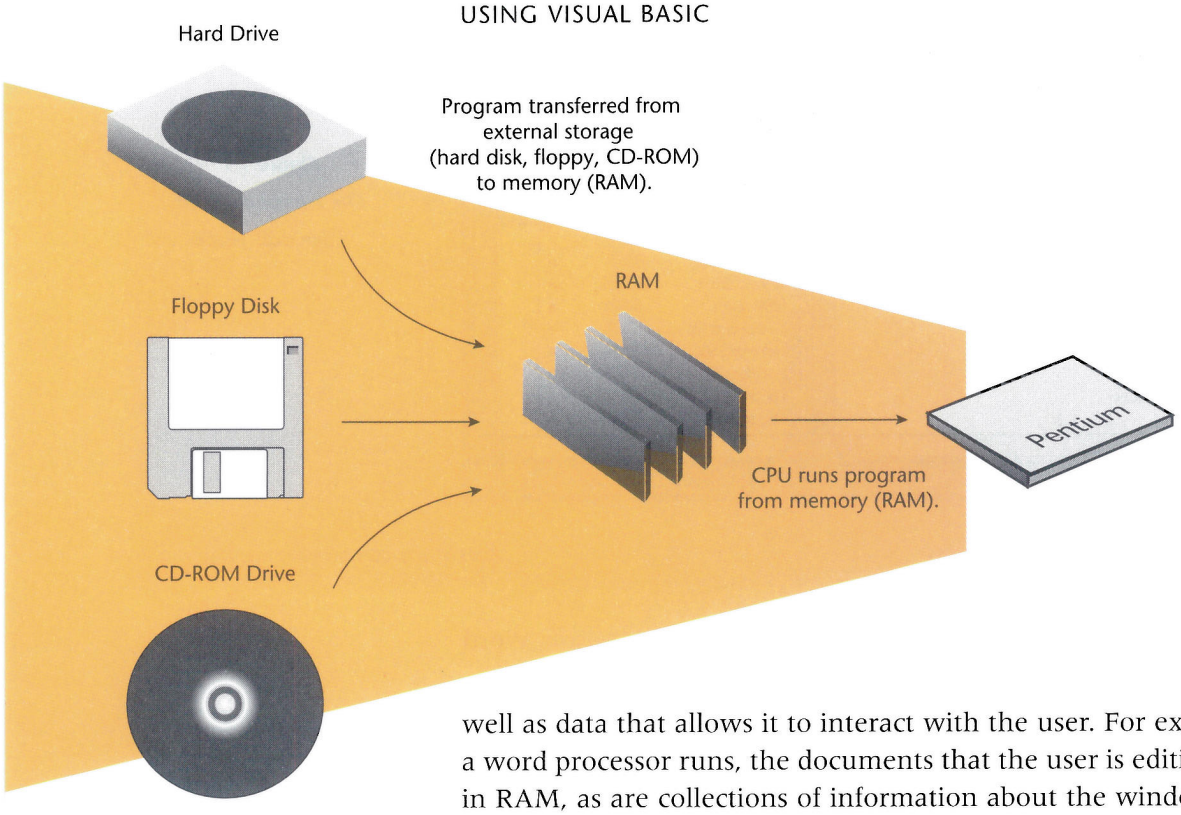


Figure 1-10
Flow of data between disk and RAM

well as data that allows it to interact with the user. For example, when a word processor runs, the documents that the user is editing are stored in RAM, as are collections of information about the windows in which those documents are displayed. Both program instructions and the data with which it works are stored in RAM.

The instructions understood by a CPU from one manufacturer are not necessary understood by a CPU from another manufacturer. A Macintosh program may not be run on an IBM-compatible PC.

QUESTIONS AND ACTIVITIES

1. MS-DOS is the Microsoft disk operating system. If you have MS-DOS installed on your computer, run **msd.com**, usually found in the DOS directory. This is the Microsoft diagnostic program. Run this program and record the following information about your computer:

   CPU type:
   Memory, Base:
   Ext:
   EMS:
   XMS:
   Video:
   Network:
   OS version:
   Mouse:
   Other Adapters:

2. If possible, find an old computer. Take the top off the machine and try to identify the CPU and memory chips. Find the video and disk controllers. Find them by tracing the connections from the montior and the disk drive to the main circuit board. Make a sketch of the inside and label what you can.

3. Using advertisements in computer magazines, or a catalog from a computer hardware vendor, design your ideal computer. List specific boards, their capabilities, and prices. Use this list as a guide.

case	motherboard, CPU
power supply	memory
disk controller	disk drives (floppy, hard)
video controller	video monitor
sound card	CD-ROM
keyboard	tape backup drive
video capture	modem
printer(s)	

4. A keyboard and a mouse are input devices. A monitor and a printer are output devices. Use a computer magazine or catalog to find and describe three input devices and three output devices you don't have on your computer.

5. Find an article about the Internet. Write a one-paragraph summary of the article.

Section

The Basics of Working in Windows

Making computers easier to use has been the vision of hardware manufacturers and software companies alike. New hardware has enabled new ways to program: more colorful graphics and faster processors let programmers use video clips; sound cards allow programs to talk or sing. The more power that's available, the more programming options are open.

Software developers have responded with ever more sophisticated ways to use the hardware. Game programmers use graphics displays in ways hardware designers could not have anticipated. The result is before you—the Windows environment. A computer environment is the sum of all the ways the computer interacts with the user. Windows is a step along the evolutionary chain of graphical user interfaces, or GUIs.

This section explains how to start Windows. You need to start Windows to run any Windows applications such as Visual Basic. It also covers how to use icons, move around in individual windows, and work with dialog boxes.

Clicking the mouse means depressing and releasing the mouse button. Double-clicking is pressing the mouse button quickly, twice. Dragging the mouse means depressing and holding the button and moving the mouse.

Starting Windows

Most personal computers start with Windows active on the screen. Sometimes, though, when you turn on a PC you will see:

```
C:\
```

This means that your computer has started in DOS, the disk operating system. The "C" refers to your hard drive, where Windows is installed. Windows is a set of programs that makes it easy to use the disk operating system. As Windows evolves, it will become an operating system. Type **win** and press Return to start Windows. The Windows Program Manager program is then run, as shown in Figure 1-11.

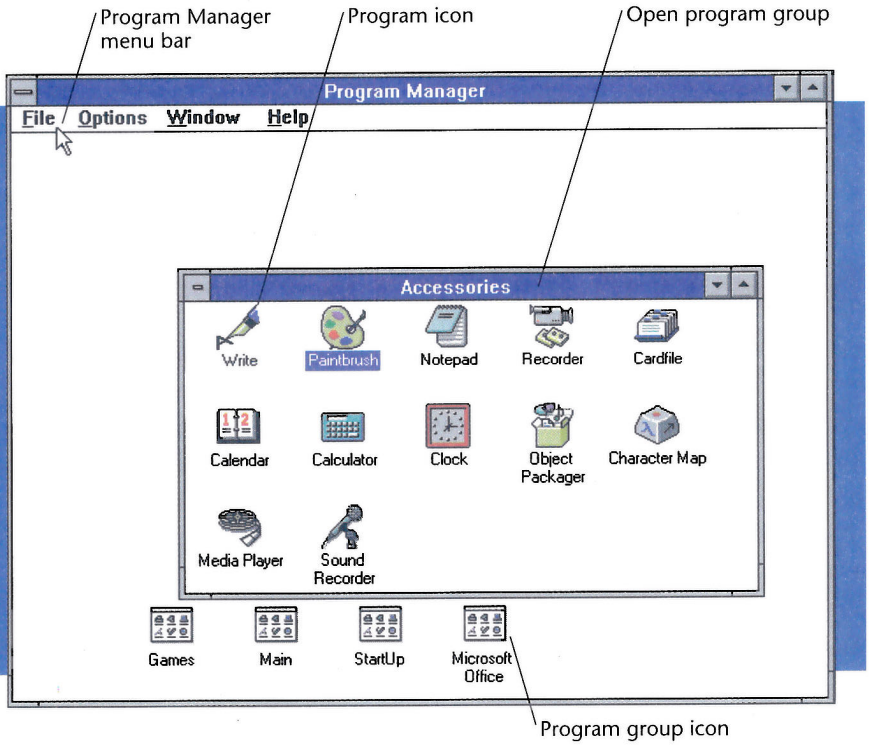


Figure 1-11
Windows Program Manager

Looking at Icons

Windows is a very graphical environment. For example, many of the choices you make in Windows are represented by icons, which are buttons with pictures on them (see Figure 1-11). You can select what you want by clicking on an icon with a mouse, rather than typing or selecting an option from a list. Icons work as shortcuts.

You can use icons to:

- Open a program or program group
- Perform a task within an application, such as cutting text

Experiment with the first of these in the Program Manager window. There you see icons for different program groups. Program groups are exactly what they sound like: groups of related programs. If you double-click on a program group icon, the program group opens. Try opening the Visual Basic program group (Figure 1-12).

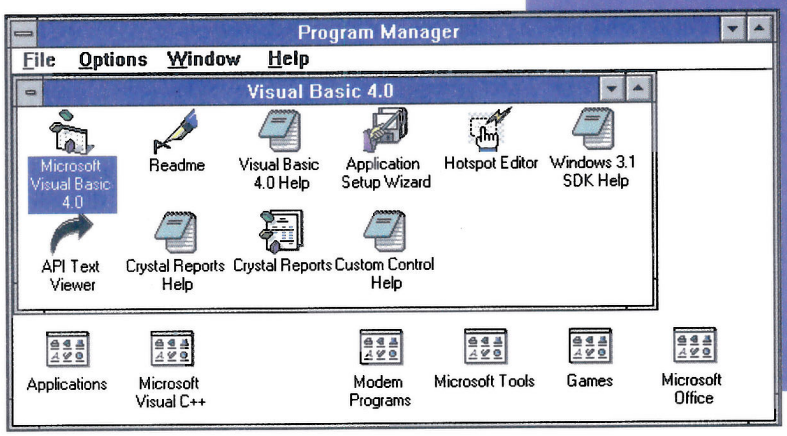


Figure 1-12
Open Visual Basic program group

With the program group open, you can see that there are icons for individual programs. To open one of these programs, you double-click on its icon.

Once you are in an application, such as Microsoft Word, you can use icons to perform many tasks. For example, if you select text in Word, then click on the Cut icon, the text is cut. Usually, these icons are grouped together in a toolbar (see Figure 1-13).

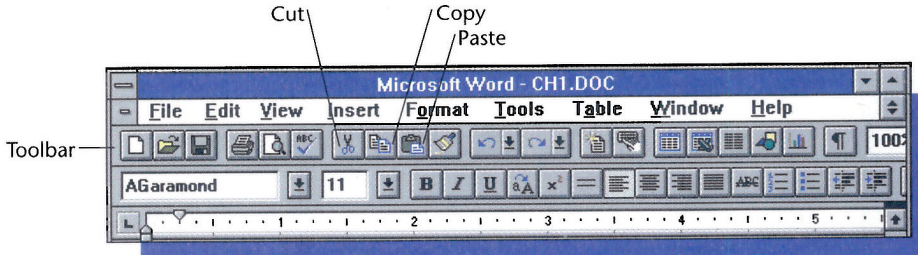


Figure 1-13
Icons in Microsoft Word

Moving Around in Windows

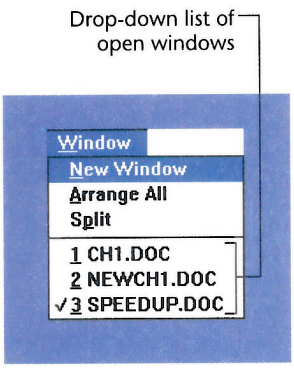


Figure 1-14
Drop-down list of open windows in Microsoft Word

In the Windows environment, you can do more than one thing at once. Each application or file that you open appears in a window. You can have multiple windows open at the same time, moving back and forth between them. You can even move material from one window to the next. Within an application, you can have more than one file open. In Figure 1-14, for example, you can see that there are three Word files open at one time.

You can also open more than one application at the same time. One way to move back and forth between open applications is to press and hold the Alt key, then press Tab.

Windows are not static on your monitor screen. You can close and open them, make them bigger and smaller, and drag them from side to side. You can use the windows in the Program Manager to experiment with the properties of windows. Windows in any Windows application work in the same way.

Double-click on an icon for a program group. The window that opens contains the icons of individual programs in the group (see Figure 1-12). The other program group icons also represent windows. These windows are said to be minimized.

Do you see the two buttons in the upper-right corner of the open program group window? If you click on the button with the down arrow, you will minimize the window. If instead you click on the button with the up arrow, the window is maximized. When a window is maximized, it takes all available space in the Program Manager window (Figure 1-15).

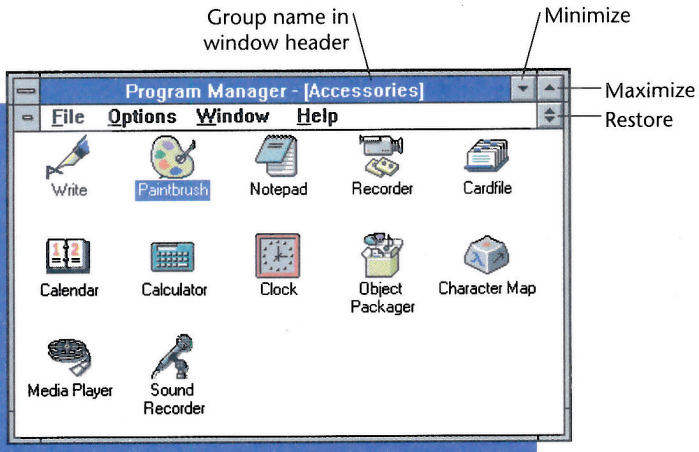


Figure 1-15
The maximized window of the Accessories program group

The Program Manager is still running, as you can see by the caption above the menu bar. In addition, a new button with both an up and a down arrow has appeared in the upper-right corner of the window. Clicking this button restores the window to its original size. Experiment with the minimize, restore, and maximize buttons for different program groups in the Program Manager (Figure 1-16).

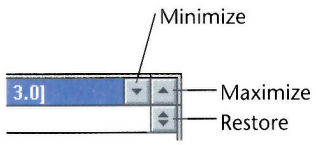


Figure 1-16
The minimize, maximize, and restore buttons

Now try resizing and moving windows by dragging the mouse. With the mouse cursor above the menu bar, on the colored caption of the window, drag the open Visual Basic window to a new position on the screen.

Now position the mouse cursor along any of the boundaries of any open window. It transforms into a double arrow, indicating you can resize the window (see Figure 1-17). If you drag on a corner of the form, you can change both the horizontal and vertical size of the window.

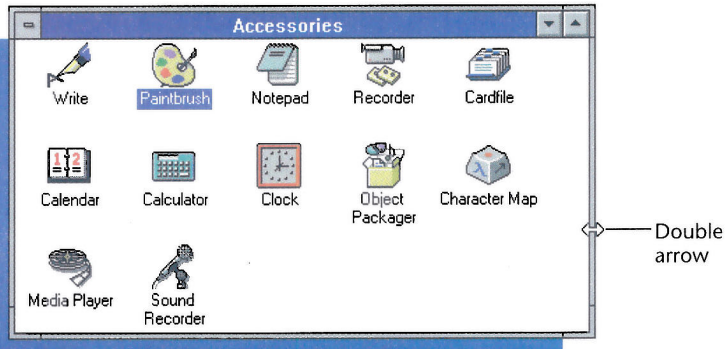


Figure 1-17
The mouse cursor as a double arrow

Notice that, in the upper-left corner of the window, there is a button with a horizontal bar in it. When you click on this button, the system menu opens (Figure 1-18). The button's icon resembles the space bar, as a reminder that you can open this menu by pressing the space bar while holding down the Alt key. You can choose to minimize, maximize, restore, resize, or move the window from this menu. Windows programs typically let you perform tasks in several different ways. As you just saw, for example, you can minimize the window by clicking on a button. You could also open this menu and select the minimize command.

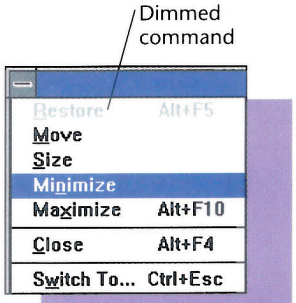


Figure 1-18
System menu

If you cannot select a command from a menu, that command is dimmed, or grayed. For instance, if you have already maximized a window before you opened this menu, the maximize command will appear dimmed. Experiment with using different commands in the system menu.

Looking at Menu Bars and Drop-Down Menus

Besides clicking on icons in a Windows application, you can also select a command from a menu. Top-level menu choices appear in a menu bar across the top of the window (see Figure 1-19). If you click on one of these, a drop-down list of commands is displayed. Clicking on a command executes it.

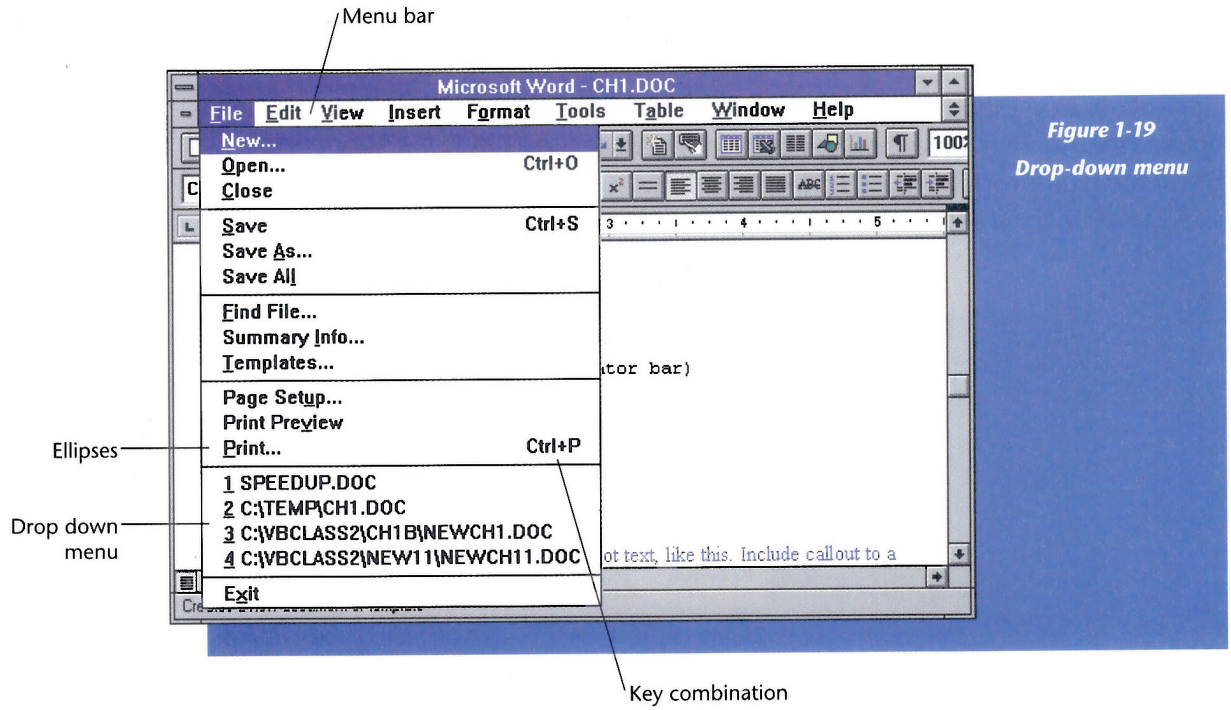


Figure 1-19
Drop-down menu

Often, a drop-down menu contains more than just the names of the commands. You may see:

- Ellipses after a command
- Names of keys (such as Ctrl+Z)
- Small triangle after a command
- Check mark

An ellipsis indicates that if you select that command, a dialog box will open. You use this dialog box to make choices about the command you are executing. See the next section for more detail on dialog boxes.

As always with Windows programs, you are given several ways to perform a task. The key combinations listed after the commands remind you that you can execute commands from the keyboard. You do not

have to go through the steps of pulling down the menu and selecting a command. The plus sign indicates that you press the two keys at the same time (the keys starting with "F" refer to the function keys along the top row of the keyboard.) Ctrl+Esc, for example, means to press and hold the Control key while pressing the Escape key.

A small triangle to the right of a command indicates that another menu opens if you select that command. You would then select a command from that menu.

A check mark indicates that a menu option is selected. For example, in Microsoft Word, you can choose to show or hide a ruler. If you have the ruler displayed, the Ruler command in the View menu is shown with a check mark. If you are looking at the list of windows you have open, then the check mark indicates the active window (see Figure 1-14).

Running an Application

Running a program from the Program Manager is easy. Find the application's icon and double-click it with the mouse. To experiment, try opening the Notepad application. This program is a simple text editor. Its editing commands are similar to those available when you are editing text in Visual Basic. Programmers use the same editing commands to manipulate the program text code. The program code is the text of the program instructions.

To run Notepad:

1. Open the Accessories program group by double-clicking on its icon (see Figure 1-20).
2. In the Accessories program group, double-click on the Notepad icon. The Notepad window opens (Figure 1-21).

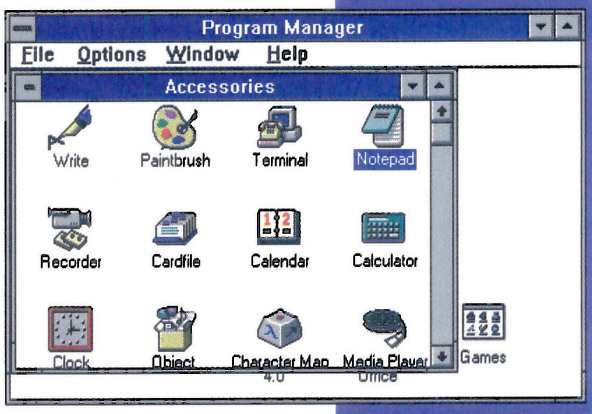


Figure 1-20
Open window for the Accessories program group

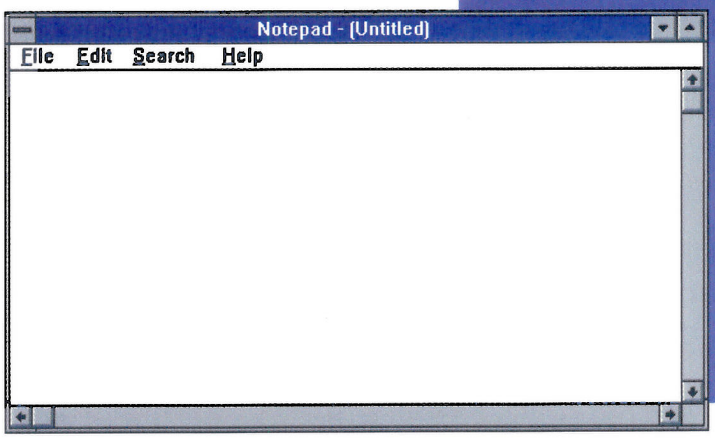


Figure 1-21
Notepad application

NAVIGATING IN A TEXT FILE

Moving around in a text file, or navigating, is easy. You can use either the keypad or the mouse. First, however, you need to enter some text with which to experiment.

1. Begin typing any text you want. Continue until the first words you typed scroll off the page to the right.
   As you can see, each line in Notepad is too long to fit in the window. To see all the text you enter, you must select the Word Wrap option.
2. Select Edit from the menu bar, then Word Wrap from the bottom of the drop-down list. What has happened to the text you entered? See Figure 1-22.
   Word Wrap is what is known as a toggle. A toggle is a command that is either on or off. When you have Word Wrap on, a check mark appears by the command.

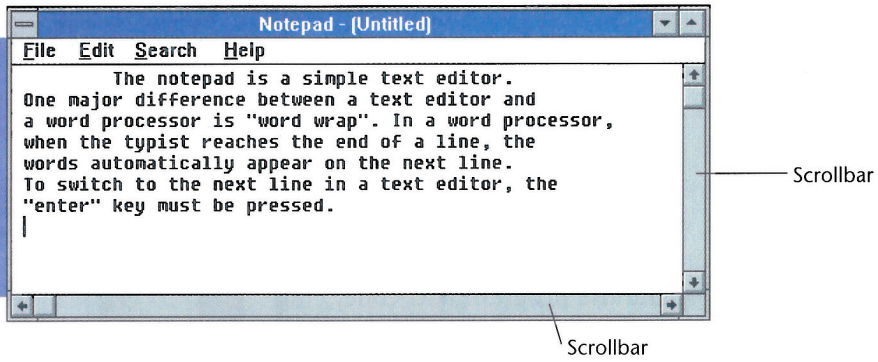


Figure 1-22
Notepad with Word Wrap selected

3. Continue typing until you have five lines of text.
4. Practice moving the text cursor with these keys and key combinations:

To move the cursor:	*Press:*
character to character	left and right arrows
word to word	Ctrl+left, Ctrl+right
line to line	up and down arrows
to top of text	Ctrl+Home
to bottom of text	Ctrl+End

SELECTING TEXT

You select text in Notepad just as you do in other Windows applications. Click at the beginning of the material you want to select and drag

the mouse to the end of the last word. The text changes color, showing it is selected.

Another way to select text is to set the cursor to the beginning of the text to be selected, press the Shift key, and move the cursor to the end. While you are holding the Shift key down, you can use any of the cursor movement keys to move the cursor. Now practice selecting different areas of the text you typed in Notepad (Figure 1-23). What happens when you try to select two separate areas of text?

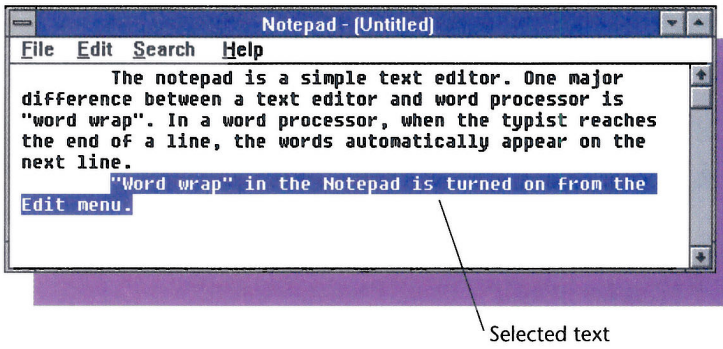


Figure 1-23
Notepad with text, some of it selected

DELETING, PASTING, AND COPYING TEXT

After you have selected text, you can delete or cut it. You should cut, not delete, any text you want to move someplace else.

1. Select a word or two of the text you entered in Notepad.
2. Press the Delete key.
   What if you change your mind? Most Windows programs let you "undo" commands such as deleting text.
3. Click on the Edit command in the menu bar, then on Undo.
   The text you deleted has been reinserted.
4. Now try cutting some text instead of deleting it. Select some text, then select Edit from the menu bar. Click on Cut, or press the keyboard combination Ctrl+X (press and hold the Control key, then press the letter "x").
5. Paste the text you cut in a different place in the text. Move the text cursor as you want. Select Edit from the menu bar, then Paste. Or, you can use the keyboard shortcut Ctrl+V.
6. You can copy text with a similar set of steps. Select the text first, then select Edit from the menu bar and Copy from the drop-down menu. Move the cursor to the new location and select Paste from the Edit drop-down menu.

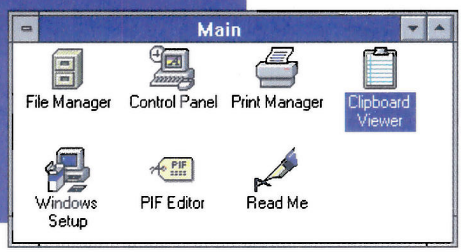
Many Windows programs have another kind of shortcut for performing common tasks such as cutting and pasting. The Word icon for pasting is shown to the left. Clicking on the icon has the same effect as selecting the command or using the keyboard combination.

When you cut or copy text, that text is moved into a special area of RAM called the Clipboard, where Windows keeps track of it. You can insert (paste) any text from the Clipboard into other places in the document you're currently working on, or for that matter, any other document in any application you're currently running. You can view the contents of the Clipboard using the Clipboard Viewer.

Now experiment with the Clipboard:

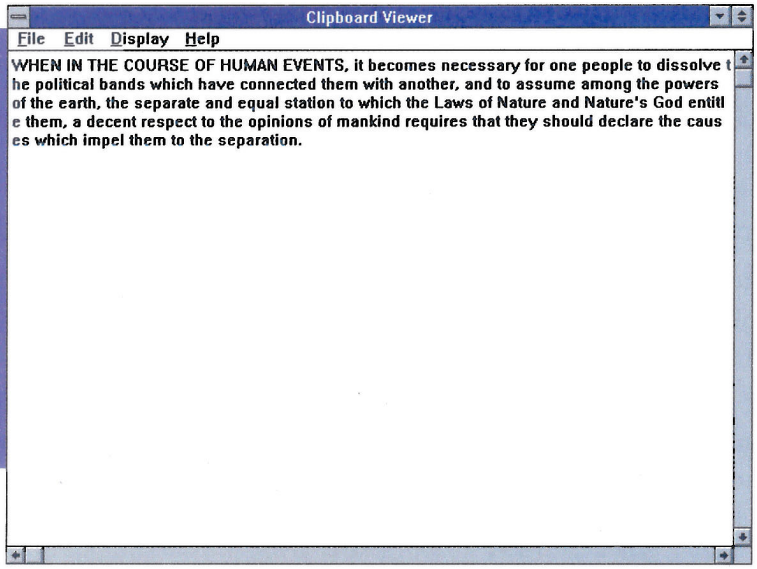
1. Cut some text from your Notepad window.
2. Open the Clipboard Viewer in the Main program group. If you have Windows for Workgroups loaded on your machine, you need to double-click on the Clipboard Viewer to open the Clipboard (Figure 1-24).

Figure 1-24 Icon for Clipboard Viewer in program group



3. Check to see that the text you cut appears in the Clipboard (Figure 1-25). Can you edit the text in the Clipboard?
4. To switch back to the Notepad, press and hold the Alt key, then press Tab.
   This is a common way of moving between windows you have open.
5. Delete some text from the Notepad using the Delete key.

Figure 1-25 Screen shot of Clipboard with text



6 Switch back to the Clipboard Viewer. Is the deleted text in the Clipboard?

7 Size and position the windows for the Notepad and the Clipboard so that both are visible. Cut text from the Notepad and watch it appear in the Viewer (Figure 1-26).

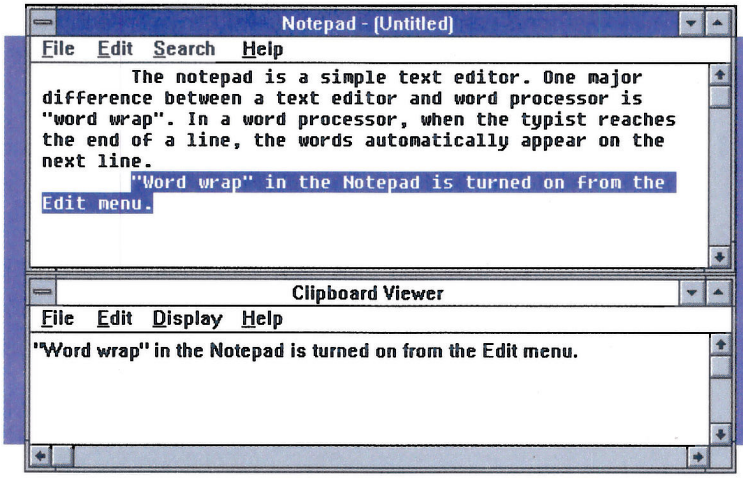


Figure 1-26
Clipboard and Notepad windows are both visible

SAVING A FILE

If you want to keep the text you have typed in the Notepad window, you must save it as a file. To do so, select File from the menu bar. The File drop-down menu is shown in Figure 1-27.

Select Save As from the drop-down menu shown in the figure. The Save As dialog box opens, as shown in Figure 1-28.

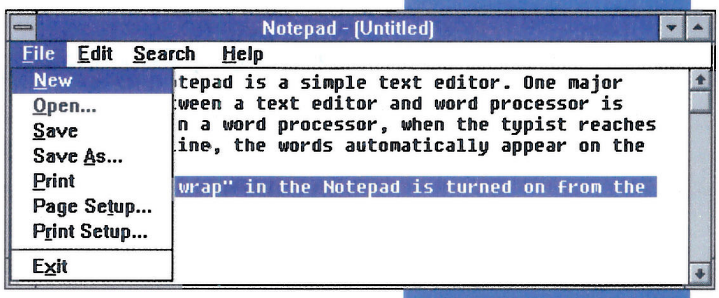


Figure 1-27
The File drop-down menu

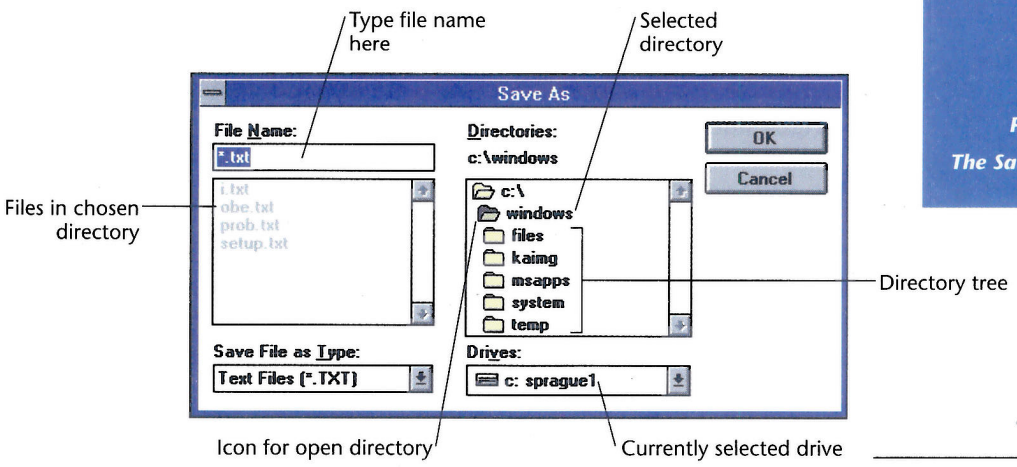


Figure 1-28
The Save As dialog box

When you save a file, you can expect you'll need to find it again later. The computer lets you organize your disk by creating directories, which you can think of as folders for keeping related files together. Within any directory, you can create other directories, called subdirectories, to store more files. Any subdirectory can have its own subdirectories. The directory that contains a particular subdirectory is called the parent directory of that subdirectory. Taken together, the whole structure of parent directories and subdirectories is called the root directory.

Now save your file:

1. Click on the Drives downward arrow. A drop-down list opens, showing the available drives. Keep the default (the C drive, or hard drive).
2. Double-click on a directory or subdirectory in the directory window. You know that you have selected a directory when it appears in the label above the window. Experiment with moving to different points in the directory tree. If you click on **C:**, you will move back up to the top of the tree.
3. Click in the File Name text box and enter a name for the file.
4. Click on the OK button to start the save.

Using the File Manager to Handle Files

Windows provides a File Manager program designed to help you manage files. With the File Manager, for example, you can create and delete directories; copy, move, rename, and delete files; and copy or format disks. You can also compare the directories from two or more disks side by side, then move or copy files between the directories.

To open the File Manager, click on the icon for the Main program group in the Program Manager window. Find the icon for the File Manager in that program group, then double-click on it. The File Manager opens, as shown in Figure 1-29.

Clicking on a directory name in the left window selects that directory. When a directory is selected, the files of that directory appear in the right side of the window.

CREATING DIRECTORIES

To create a directory, use the vertical scrollbar to move up or down the directory tree in the left side of the window. Find the parent for the directory you want to create. For example, if you have a directory named fruit and you want to create a subdirectory named apples, you would click on the fruit directory. If you select the top line of the directory (**C:**), this is the root directory of the drive.

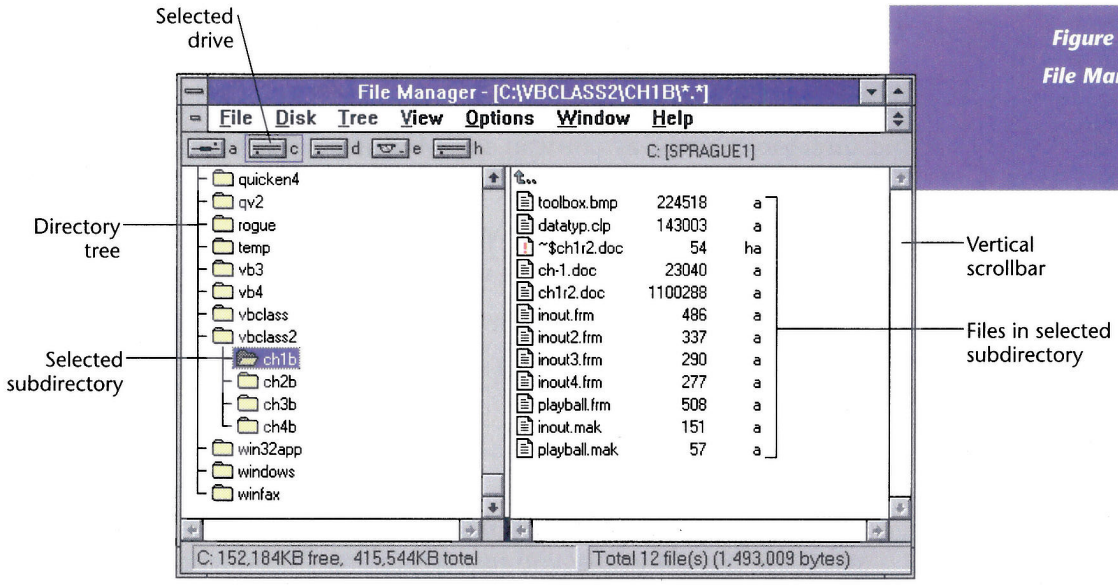


Figure 1-29
File Manager

From the File menu, select Create Directory. A Create Directory dialog box opens, as shown in Figure 1-30.

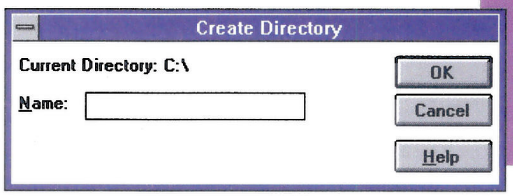


Figure 1-30
Create Directory dialog box

Enter the name of the directory in the Name text box and click OK. This new directory will appear in the directory window of the File Manager. Now try creating a directory for the Notepad file you created earlier in this chapter. Try to pick a descriptive name, such as Notefile. Windows will cut the extra letters off any name longer than eight characters (Figure 1-31).

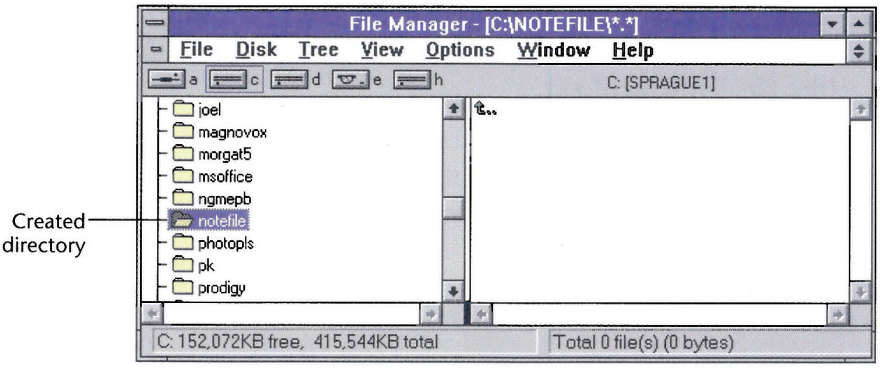


Figure 1-31
Created directory

MOVING FILES BETWEEN DIRECTORIES

After you have created a directory, you can experiment with moving a file or two into it. Select the directory you just created; it is empty of file and directory names, as you can see from the right window. To move the Notepad file into the new directory:

1. Select Window from the File Manager menu bar, then select New Window from the drop-down menu.
2. Select Window again, then click on Tile. You will now see two copies of the file and directory windows of the File Manager (see Figure 1-32).
3. In one of the copies, select the directory in which you originally stored the Notepad file. In the example shown in Figure 1-32, the Notepad file was saved in the Windows directory. In the other copy, select the new directory you just created for the Notepad files.

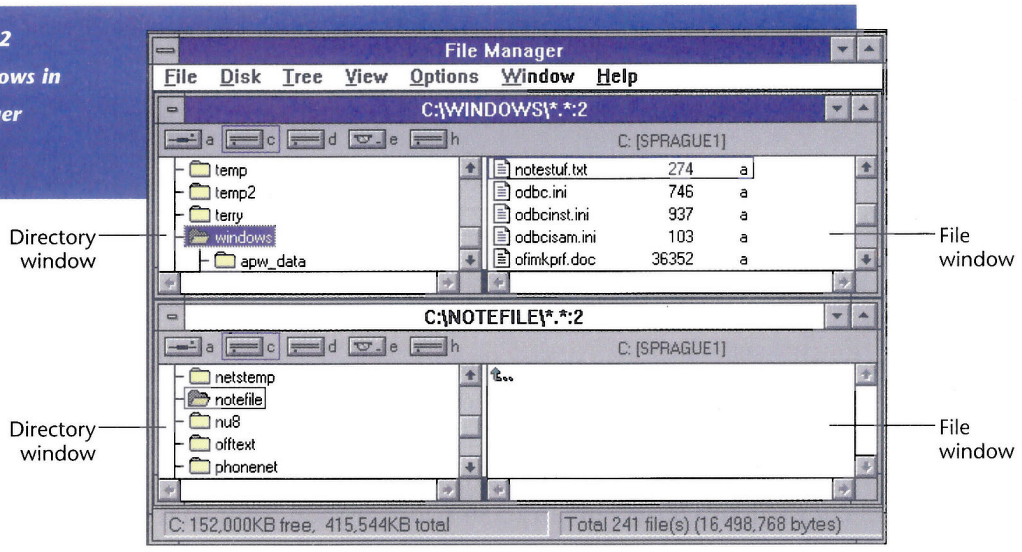


Figure 1-32
Copies of windows in File Manager

4. Find the Notepad file in its original directory and select it by clicking on its name. In the example shown in the figure, the name of the file is **notestuf.txt**.
5. Drag the file to the window displaying the directory you just created (notefile directory, in this example). Connected to the cursor as you move it will be an icon of a sheet of paper, representing the file (Figure 1-33).

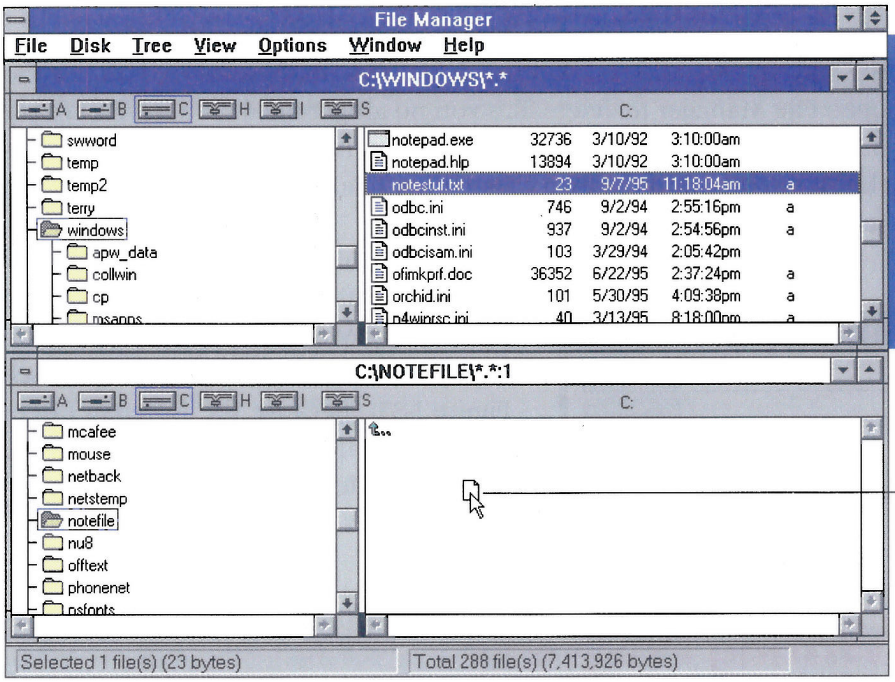


Figure 1-33
Moving a file

6 When a message box appears asking if you are sure you want to move the file, check the file names and then click on Yes.
The File Manager moves the file from one directory to the other (see Figure 1-34).

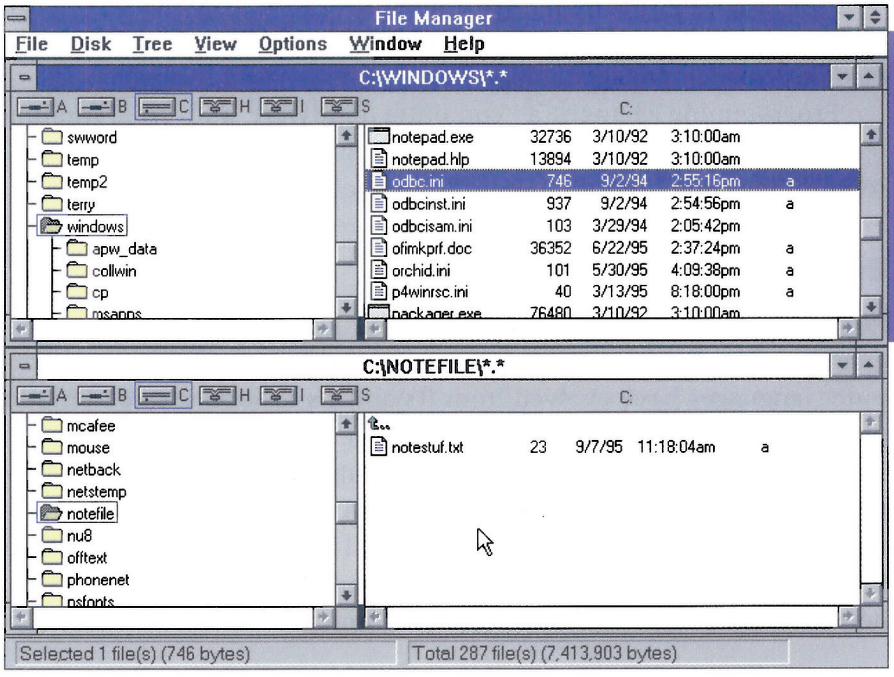


Figure 1-34
Moved file

DELETING FILES AND DIRECTORIES

In some applications, you cannot delete files. In these cases, you need to open File Manager to delete files you no longer need. You have a couple of options for doing so. Find the file, select it with the mouse, and press the Delete key. You can also click on the File Manager's File menu, then select Delete from the drop-down menu.

If you select a directory instead of a file, pressing the Delete key deletes the directory. For both files and directories, Windows prompts you to be sure you want to make the deletion (see Figure 1-35). Because the Notepad directory you created was just an experiment, you should delete it now.

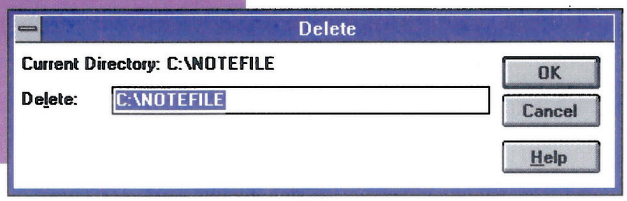


Figure 1-35 Deleting from the File Manager

4

Section

Writing Programs

Why write computer programs? Programming computers is fun. Writing programs that control expensive hardware, that turn a dumb machine into a homework helper or playmate, is tremendously satisfying. Although learning to program entails a certain amount of work, that work is repaid through a real feeling of accomplishment.

Writing computer programs is good business. Whether you write programs professionally, or occasionally write a program to solve a business or personal management problem, programming means money.

Programming changes as computer hardware and programming techniques evolve. A typical PC today is vastly more powerful than any room-filling computer of 20 years ago and costs dramatically less.

Evolution of Programming

Programming means communicating with a machine. Computer languages define the symbols and structures used to communicate. Computer languages have evolved from 0's and 1's to sophisticated English phrases. This section reviews that evolution.

At the most basic level, the CPU of a computer understands instructions coded in the binary number system. As noted earlier in the chapter, this system uses only the digits 0 and 1 to represent values. These instructions are stored in memory (RAM) and are sequentially fetched, decoded, and executed by the processing unit.

In the earliest computers, these codes were entered with on/off switches on the front panel of the computer. Long rows of switches were used to represent bits. If the switch was in the up position, it represented a 0. If the switch were down, it represented a 1. A pushbutton indicated the number was set. When the pushbutton was pressed, the switches were read and the 0's and 1's were transferred to RAM. Accurately entering the long strings of zeros and ones was very difficult.

Early in the development of these machines, the computer itself was used to help interpret instructions. Instead of simple on/off switches, pushbuttons representing the digits of base 8 or base 16 numbers were used. The computer converted the values represented by the pushbuttons to their binary equivalents.

Programming began evolving when the binary number system was replaced with systems that were easier for people to understand. Imagine trying to read or remember a sequence of 0s and 1s! Binary numbers, still used internally, were replaced externally with numbers based on powers of 8 or 16. These values were shorter and easier to use. Programmers needed a more compact notation in which to work. When these more concise programs were entered into the computer, the computers translated them back into binary and then executed the result. Programs still consisted of numbers, though.

Large computers used more and more sophisticated languages starting in the 1940s. By the time the 1950s were over, a number of easy-to-use computer languages were in use. Small computers didn't arrive on the scene until the first years of the 1970s. The evolution of these small computers was fairly rapid. By the late 1970s, a number of computer languages were available for microcomputers.

The evolution continued toward a computer language that was easier to understand and remember. Numeric codes were replaced with short descriptive syllables called mnemonics. These short syllables were easier to remember (and type) than the longer commands that they represent. Examples include:

```
ADD, ADDC, MOV, SUB, SUBC, ASL, ROT, LDA
```

Each of these instructions is a command that the processor can translate to binary, understand, and execute. By itself, one of these commands can accomplish very little. When joined together, though, these commands can perform any operation of which the computer is capable. For instance, ADDC is the Add with Carry command. This adds two numbers including the carry left over from a previous operation.

The languages that use these short syllables to create code are called assembly languages. Each different processor (such as the one for a PC or a Macintosh) has its own unique assembly language. The languages are different because each kind of processor has a unique architecture that is embodied in its assembly language. A PC CPU cannot understand a program written in the Macintosh assembly language. In hindsight, these assembly languages are called low-level computer languages.

Assembly languages greatly simplified programming, but there was still a long way to go. Computers could understand these languages well enough—much easier than the programmers themselves. Once again, computers were asked to aid in the translation of commands more easily understood by humans, and high-level computer languages were born. Complex computer programs are used to translate the statements of computer languages to the binary commands understood by the processor. The programs are called compilers. The first of these languages, developed in the 1950s, are still in use today.

High-level languages use words similar to English words, and operations similar to familiar math operations to express computer operations. "Visual" Basic advances the evolution of computer languages by introducing a strong graphic element into programming. Programmers work with icons that represent parts of a Visual Basic program.

Machine	Base 16	Assembly	High Level
`1101 1001`	`D9`	`INC`	`X = X + 1`

INC is the mnemonic for increment.

Models of Programming

As programming languages have evolved, so have methods of programming. This section introduces traditional procedural programming and the newer event-driven programming. The section uses analogies to illustrate the differences.

Imagine that you play in the school band and you are preparing for a concert. That preparation includes learning the music for the pieces to be performed. As you practice each piece, the teacher leads the musicians through section after section, repeating each until it's just right. Finally, the conductor leads the band through an entire performance from beginning to end.

Imagine now that you are a goalie for a soccer team. Your training includes physical conditioning, drills and hours of practice in goal responding to shots. A game consists of reacting, with practiced responses, to the events of the game—the shots on goal.

PROCEDURAL PROGRAMMING

Procedural programming is like playing a piece of music. Each program has a beginning, a middle, and an end, and a particular set of steps to get from one place to another. In procedural programming, the programmer defines the path of the information entered into the program from the beginning of the program, step by step, to the end of the program. Throughout the entire process the program is in control. The program controls the gathering of data from the keyboard or from a file of information, processing the data in some way, and then preparing the information for output or display.

The beginning, middle, and end of a program are not clearly separated, but they can be summarized as shown in the list below.

1. Entering data (beginning)
2. Processing data (middle)
3. Displaying information (end)

For instance, a program to calculate the price of going to a prom, written in the old procedural way, could be summarized as follows:

1. Collect information about prices
2. Calculate totals
3. Display the results

Just like a musician practicing a piece of music, the programmer breaks this program into parts and solves each part, combining the parts into final form when each is mastered.

EVENT-DRIVEN PROGRAMMING

Event-driven programming is like playing goalie in a soccer game. Like a goalie, a program responds to events: a keypress or a mouse click.

A programmer using the event-driven model of programming will use many of the same step-by-step solutions used by a programmer following the procedural model. The techniques are different, though, because the programmer writes programs in which the user is in control. In this model of programming, it is the events generated by the computer user that control the flow of the program.

Visual Basic uses the event-driven model of programming. Programmers in Visual Basic design programs in an environment that responds to user-initiated events.

QUESTIONS AND ACTIVITIES

1. Describe three uses for a computer in the home. Be specific; "Doing your homework" is not a good answer.
2. Give the step-by-step instructions required to open a can of frozen juice and make a pitcher of juice. Is this activity procedural or event-driven?
3. Give three examples of jobs where a day's work cannot be completely planned or anticipated. In such a job, it is important to respond to events as they occur.
4. For one of the jobs listed above describe what events might occur and what a proper response to the events might be.
5. Describe what events might occur when driving a car to school. Describe some possible responses to those events.

Summary

Visual Basic is used to provide solutions for the needs of home, business, industry, and the scientific community.

Visual Basic is used to interact and control other applications, like Microsoft Access, Excel, and Word.

The CPU is the central processing unit of the computer. The CPU runs programs and manipulates data.

RAM is random access memory. This memory holds programs as they run and data as it is processed.

ROM, or read-only memory, permanently stores programs that boot up the computer.

Memory records information using the binary number system, a system that uses only 0's and 1's.

Each location in memory has an address, sometimes expressed in binary.

A floppy disk, hard drive, and CD-ROM disk are examples of external storage.

Video and sound cards, printers, scanners, and modems are examples of peripheral devices—devices connected to the computer.

Loading programs means copying programs from external storage to RAM. Running programs means executing a program line by line from memory by the CPU.

Windows can be minimized, maximized, and, once altered, restored.

Windows can be moved and resized with mouse clicks and drags.

Text is manipulated with mouse clicks, menu choices, or keyboard commands. Text can be copied, cut, and pasted.

Files are manipulated with the Program and File Manager programs. Directories can be created and removed. Files can be created, deleted, copied, and moved.

Computer languages have evolved from languages best understood by machines, or low-level languages, to languages easily understood by people, or high-level languages.

Procedural programming assumes every problem can be solved with a predictable step-by-step solution. The solution is always under machine control.

Event-driven programming solves problems by handling events caused by the user and the computer environment. Many events are user-initiated.


txtLastName
txt
txtLastName
cmdDis
txtLastName
UnpaidBalance
lblBirthplace
DisplayPicture