	Read	Write/Delete	Execute
Owner	Yes	Yes	No
Group	Yes	No	No
World	No	No	No

Suppose that this grid represents the permissions on a data file used in project Alpha. The owner of the file (perhaps the manager of the project) may read from or write to the file. Suppose also that the owner sets up a group (using the operating system) called TeamAlpha, which contains all members of the project team, and associates that group with this data file. The members of the TeamAlpha group may read the data in the file, but may not change it. No one else is given any permission to access the file. Note that no user is given execute privileges for the file because it is a data file, not an executable program.

Other operating systems set up their protection schemes in different ways, but the goal is the same: to control access so as to protect against deliberate attempts to gain inappropriate access as well as minimize inadvertent problems caused by well-intentioned but hazardous users.

11.2 Directories

As mentioned earlier, a directory is a named collection of files. It is a way to group files so that you can organize them in a logical manner. For example, you might place all of your papers and notes for a particular class in a directory created for that class. The operating system must carefully keep track of directories and the files they contain.

A directory, in most operating systems, is represented as a file. The directory file contains data about the other files in the directory. For any given file, the directory contains the file name, the file type, the address on disk where the file is stored, and the current size of the file. The directory also contains information about the protections set up for the file. In addition, it may hold information describing when the file was created and when it was last modified.

The internal structure of a directory file could be set up in a variety of ways, and we won't explore those details here. However, once it is set up, this structure must be able to support the common operations that are performed on directory files. For instance, the user must be able to list all of the files in the directory. Other common operations are creating,

?

RFID tags

As you leave a store after buying a pack of batteries, the batteries "tell" the store's inventory system to order batteries because stock is low. Radio-frequency identification (RFID) makes this type of communication possible. If the battery packaging has an RFID tag, it can tell a central radio-frequency transceiver where it is. In addition to items in retail stores, RFID technology is used to track shipping pallets, library books, vehicles, and animals. If you've ever used EZPass to go through a toll booth, or SpeedPass® to pay for your gas, you've used RFID technology. Researchers have even experimented with implanting RFID tags in people! In 2004, a club owner in Barcelona, Spain, and Rotterdam, the Netherlands, offered to implant his VIP customers with RFID tags. These chips identified the customers as VIPs, and the chips were used by the customers to pay for their drinks.

deleting, and renaming files within a directory. Furthermore, the directory is often searched to see whether a particular file is in the directory.

Another key issue when it comes to directory management is the need to reflect the relationships among directories, as discussed in the next section.

Directory Trees

A directory of files can be contained within another directory. The directory containing another directory is usually called the parent directory, and the one inside is called a subdirectory. You can set up such nested directories as often as needed to help organize the file system. One directory can contain many subdirectories. Furthermore, subdirectories can contain their own subdirectories, creating a hierarchy structure. To visualize this hierarchy, a file system is often viewed as a directory tree, showing directories and files within other directories. The directory at the highest level is called the root directory.

» Directory tree A structure showing the nested directory organization of the file system

» Root directory The topmost directory, in which all others are contained

» Working directory The currently active subdirectory

As an example, consider the directory tree shown in Figure 11.4. This tree represents a very small part of a file system that might be found on a computer using some flavor of the Microsoft® Windows® operating system. The root of the directory system is referred to using the drive letter `C:` followed by the backslash (`\`).

In this directory tree, the root directory contains three subdirectories: `WINDOWS`, `My Documents`, and `Program Files`. Within the `WINDOWS` directory, there is a file called `calc.exe` as well as two other subdirectories (`Drivers` and `System`). Those directories contain other files and subdirectories. Keep in mind that all of these directories in a real system would typically contain many more subdirectories and files.

Personal computers often use an analogy of folders to represent the directory structure, which promotes the idea of containment (folders inside other folders, with some folders ultimately containing documents or other data). The icon used to show a directory in the graphical user interface of an operating system is often a graphic of a manila file folder such as the kind you would use in a physical file drawer.

In Figure 11.4, two files have the name `util.zip` (in the `My Documents` directory, and in its subdirectory called `downloads`). The nested directory structure allows for multiple files to have the same name. All the files in any one directory must have unique names, but files in different directories or subdirectories can have the same name. These files may or may not contain the same data; all we know is that they have the same name.

At any point in time, you can be thought of as working in a particular location (that is, a particular subdirectory) of the file system. This subdirectory is referred to as the current working directory. As you "move" around in the file system, the current working directory changes.

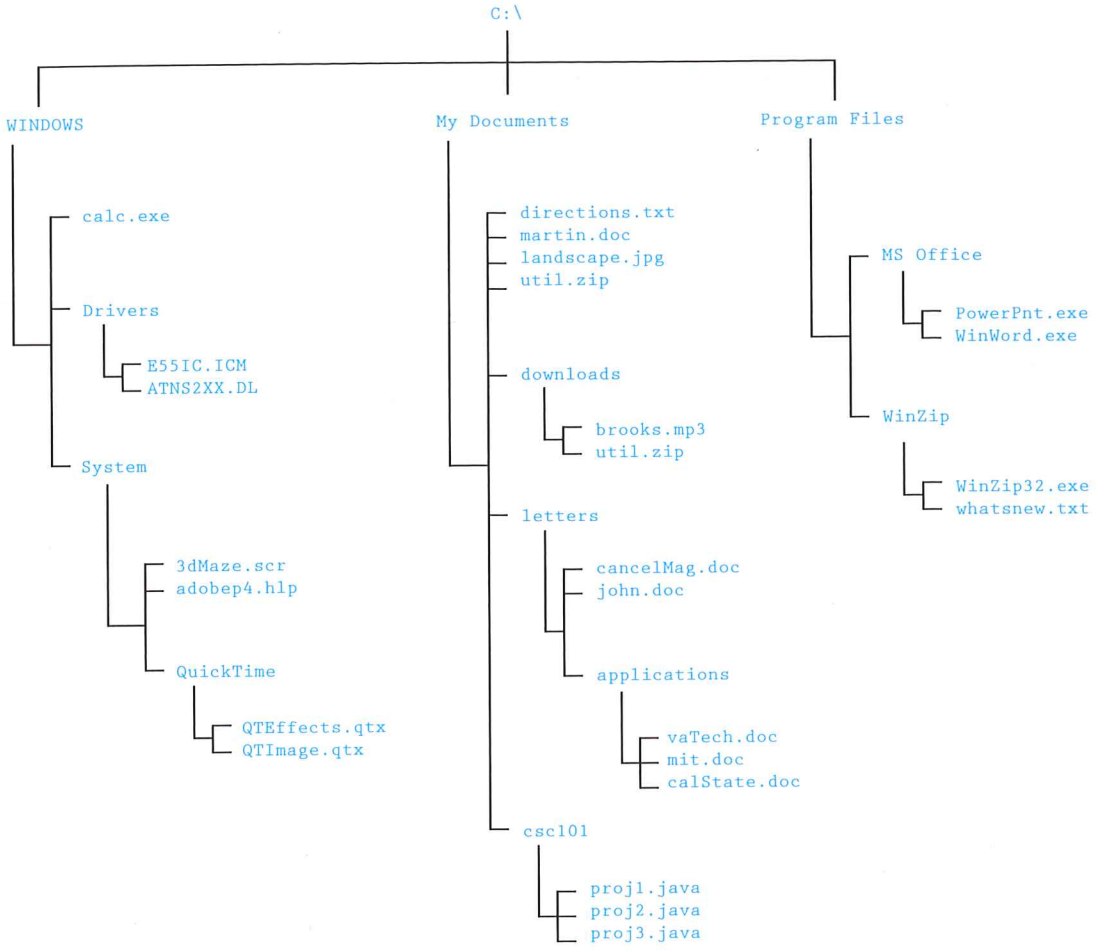


FIGURE 11.4 A Windows directory tree

The directory tree shown in Figure 11.5 is representative of one from a UNIX file system. Compare and contrast it to the directory tree in Figure 11.4. Both show the same concepts of subdirectory containment, but the naming conventions for files and directories are different. UNIX was developed as a programming and system-level environment, so it uses much more abbreviated and cryptic names for directories and files. Also, in a UNIX environment, the root is designated using a forward slash (/).

Path Names

How do we specify one particular file or subdirectory? Well, there are several ways to do it.

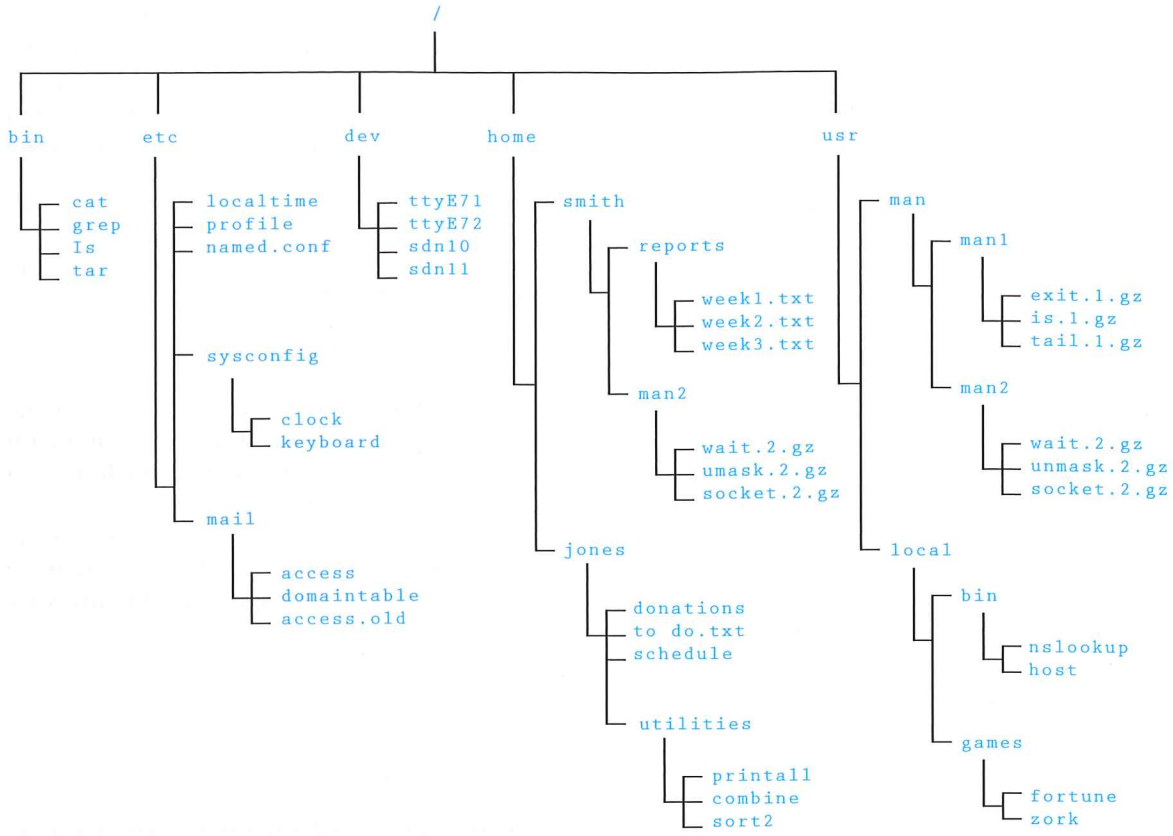


FIGURE 11.5 A UNIX directory tree

If you are working in a graphical user interface to the operating system, you can double-click with your mouse to open a directory and see its contents. The active directory window shows the contents of the current working directory. You can continue "moving" through the file system by using mouse clicks, thereby changing the current working directory, until you find the desired file or directory. To move up the directory structure, you can often use an icon on the window bar or a pop-up menu option to move to the parent directory.

Most operating systems also provide a nongraphical (text-based) interface to the operating system. Therefore, we must be able to specify file locations using text. This ability is very important for system instructions stored in operating system batch command files. Commands such as `cd` (which stands for "change directory") can be used in text mode to change the current working directory.

» **Path** A text designation of the location of a file or subdirectory in a file system

» **Absolute path** A path that begins at the root and includes all successive subdirectories

» **Relative path** A path that begins at the current working directory

To indicate a particular file using text, we specify that file's path, which is the series of directories through which you must go to find the file. A path may be absolute or relative. An absolute path name begins at the root and specifies each step down the tree until it reaches the desired file or directory. A relative path name begins from the current working directory.

Let's look at examples of each type of path. The following are absolute path names based on the directory tree shown in Figure 11.4:

```
C:\Program Files\MS Office\WinWord.exe
C:\My Documents\letters\applications\vaTech.doc
C:\Windows\System\QuickTime
```

Each path name begins at the root and proceeds down the directory structure. Each subdirectory is separated by the backslash (\). Note that a path can specify a specific document (as it does in the first two examples) or an entire subdirectory (as it does in the third example).

Absolute paths in a UNIX system work the same way, except that the character used to separate subdirectories is the forward slash (/). Here are some examples of absolute path names that correspond to the directory tree in Figure 11.5:

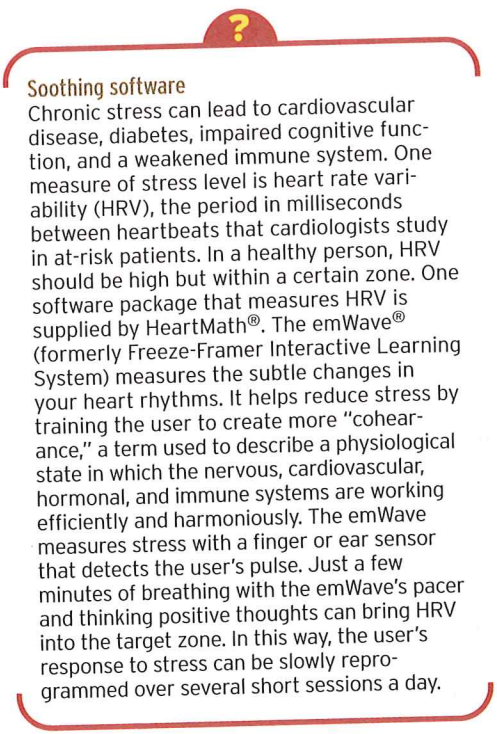
```
/bin/tar
/etc/sysconfig/clock
/usr/local/games/fortune
/home/smith/reports/week1.txt
```

Relative path names are based on the current working directory. That is, they are relative to your current position (hence the name). Suppose the current working directory is `C:\My Documents\letters` (from Figure 11.4). Then the following relative path names could be used:

```
cancelMag.doc
applications\calState.doc
```

The first example specifies just the name of the file, which can be found in the current working directory. The second example specifies a file in the `applications` subdirectory. By definition, the first part of any valid relative path is located in the working directory.

Sometimes when using a relative path we need to work our way back up the tree. Note that this consideration was not an issue when we used absolute paths. In most operating systems, two dots (. .) are used to specify the parent directory (a single dot is used to specify the current working directory). Therefore,



Soothing software

Chronic stress can lead to cardiovascular disease, diabetes, impaired cognitive function, and a weakened immune system. One measure of stress level is heart rate variability (HRV), the period in milliseconds between heartbeats that cardiologists study in at-risk patients. In a healthy person, HRV should be high but within a certain zone. One software package that measures HRV is supplied by HeartMath®. The emWave® (formerly Freeze-Framer Interactive Learning System) measures the subtle changes in your heart rhythms. It helps reduce stress by training the user to create more "coherearance," a term used to describe a physiological state in which the nervous, cardiovascular, hormonal, and immune systems are working efficiently and harmoniously. The emWave measures stress with a finger or ear sensor that detects the user's pulse. Just a few minutes of breathing with the emWave's pacer and thinking positive thoughts can bring HRV into the target zone. In this way, the user's response to stress can be slowly reprogrammed over several short sessions a day.

if the working directory is `C:\My Documents\letters`, the following relative path names are also valid:

```
..\landscape.jpg
..\csc111\proj2.java
..\..\WINDOWS\Drivers\E55IC.ICM
..\..\Program Files\WinZip
```

UNIX systems work essentially the same way. Using the directory tree in Figure 11.5, and assuming that the current working directory is `/home/jones`, the following are valid relative path names:

```
utilities/combine
../smith/reports
../../dev/ttyE71
../../usr/man/man1/ls.1.gz
```

Most operating systems allow the user to specify a set of paths that are searched (in a specific order) to help resolve references to executable programs. Often that set of paths is specified using an operating system variable called `PATH`, which holds a string that contains several absolute path names. Suppose, for instance, that user `jones` (from Figure 11.5) has a set of utility programs that he uses from time to time. They are stored in the directory `/home/jones/utilities`. When that path is added to the `PATH` variable, it becomes a standard location used to find programs that `jones` attempts to execute. Therefore, no matter what the current working directory is, when `jones` executes the `printall` program (just the name by itself), it is found in his utilities directory.

11.3 Disk Scheduling

The most important hardware device used as secondary memory is the magnetic disk drive. File systems stored on these drives must be accessed in an efficient manner. It turns out that transferring data to and from secondary memory is the worst bottleneck in a general computer system.

Recall from Chapter 10 that the speed of the CPU and the speed of main memory are much faster than the speed of data transfer to and from secondary memory such as a magnetic disk. That's why a process that must perform I/O to disk is made to wait while that data is transferred, to give another process a chance to use the CPU.

Because secondary I/O is the slowest aspect of a general computer system, the techniques for accessing data on a disk drive are of crucial importance to file systems. As a computer deals with multiple processes over a period of time, requests to access the disk accumulate. The technique that the operating