2

NAVIGATION

The first thing we need to learn (besides how to type) is how to navigate the file system on our Linux system. In this chapter, we will introduce the following commands:

`pwd` Print name of current working directory

`cd` Change directory

`ls` List directory contents

Understanding the File System Tree

Like Windows, a Unix-like operating system such as Linux organizes its files in what is called a *hierarchical directory structure*. This means they are organized in a tree-like pattern of directories (sometimes called *folders* in other systems), which may contain files and other directories. The first directory in the file system is called the *root directory*. The root directory contains files and subdirectories, which contain more files and subdirectories, and so on.

Note that unlike Windows, which has a separate file system tree for each storage device, Unix-like systems such as Linux always have a single file system tree, regardless of how many drives or storage devices are at-

tached to the computer. Storage devices are attached (or more correctly, *mounted*) at various points on the tree according to the whims of the *system administrator*, the person (or people) responsible for the maintenance of the system.

The Current Working Directory

Most of us are probably familiar with a graphical file manager that represents the file system tree, as illustrated in Figure 2-1.

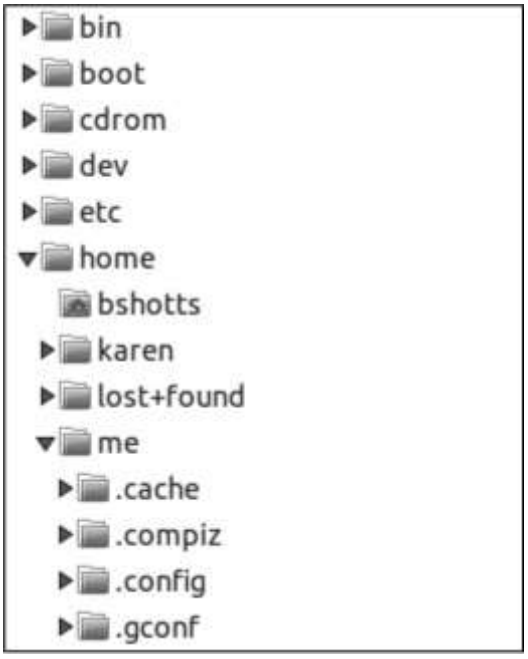

Figure 2-1: File system tree as shown by a graphical file manager

Notice that the tree is usually shown upended, that is, with the root at the top and the various branches descending below.

However, the command line has no pictures, so to navigate the file system tree, we need to think of it in a different way.

Imagine that the file system is a maze shaped like an upside-down tree and we are able to stand in the middle of it. At any given time, we are inside a single directory, and we can see the files contained in the directory and the pathway to the directory above us (called the *parent directory*) and any subdirectories below us. The directory we are standing in is called the *current working directory*. To display the current working directory, we use the `pwd` (print working directory) command.

```
[me@linuxbox ~]$ pwd
/home/me
```

When we first log in to our system (or start a terminal emulator session), our current working directory is set to our *home directory*. Each user account is given its own home directory, and it is the only place a regular user is allowed to write files.

Listing the Contents of a Directory

To list the files and directories in the current working directory, we use the `ls` command.

```
[me@linuxbox ~]$ ls
Desktop  Documents  Music  Pictures  Public  Templates  Videos
```

Actually, we can use the `ls` command to list the contents of any directory, not just the current working directory, and there are many other fun things it can do as well. We'll spend more time with `ls` in Chapter 3.

Changing the Current Working Directory

To change our working directory (where we are standing in the tree-shaped maze), we use the `cd` command. To do this, type `cd` followed by the pathname of the desired working directory. A pathname is the route we take along the branches of the tree to get to the directory we want. We can specify pathnames in one of two different ways: as *absolute pathnames* or as *relative pathnames*. Let's deal with absolute pathnames first.

Absolute Pathnames

An absolute pathname begins with the root directory and follows the tree branch by branch until the path to the desired directory or file is completed. For example, there is a directory on your system in which most of the system's programs are installed. The directory's pathname is */usr/bin*. This means from the root directory (represented by the leading slash in

the pathname) there is a directory called *usr* that contains a directory called *bin*.

```
[me@linuxbox ~]$ cd /usr/bin
[me@linuxbox bin]$ pwd
/usr/bin
[me@linuxbox bin]$ ls
...Listing of many, many files ...
```

Now we can see that we have changed the current working directory to */usr/bin* and that it is full of files. Notice how the shell prompt has changed? As a convenience, it is usually set up to automatically display the name of the working directory.

Relative Pathnames

Where an absolute pathname starts from the root directory and leads to its destination, a relative pathname starts from the working directory. To do this, it uses a couple of special notations to represent relative positions in the file system tree. These special notations are . (dot) and .. (dot dot).

The . notation refers to the working directory, and the .. notation refers to the working directory's parent directory. Here is how it works. Let's change the working directory to */usr/bin* again.

```
[me@linuxbox ~]$ cd /usr/bin
[me@linuxbox bin]$ pwd
/usr/bin
```

Now let's say that we wanted to change the working directory to the parent of */usr/bin*, which is */usr*. We could do that two different ways, either with an absolute pathname:

```
[me@linuxbox bin]$ cd /usr
[me@linuxbox usr]$ pwd
/usr
```

or with a relative pathname:

```
[me@linuxbox bin]$ cd ..
[me@linuxbox usr]$ pwd
/usr
```

Two different methods with identical results. Which one should we use? The one that requires the least typing!

Likewise, we can change the working directory from */usr* to */usr/bin* in two different ways, either using an absolute pathname:

```
[me@linuxbox usr]$ cd /usr/bin
[me@linuxbox bin]$ pwd
/usr/bin
```

or using a relative pathname:

```
[me@linuxbox usr]$ cd ./bin
[me@linuxbox bin]$ pwd
/usr/bin
```

Now, there is something important to point out here. In almost all cases, we can omit the `./` part because it is implied. Typing the following does the same thing:

```
[me@linuxbox usr]$ cd bin
```

In general, if we do not specify a pathname to something, the working directory will be assumed.

IMPORTANT FACTS ABOUT FILENAMES

On Linux systems, files are named in a manner similar to that of other systems such as Windows, but there are some important differences.

- Filenames that begin with a period character are hidden. This only means that `ls` will not list them unless you say `ls -a`. When your account was created, several hidden files were placed in your home directory to configure things for your account. In Chapter 11 we will take a closer look at some of these files to see how you can customize your environment. In addition, some applications place their configuration and settings files in your home directory as hidden files.
- Filenames and commands in Linux, like Unix, are case sensitive. The filenames *File1* and *file1* refer to different files.
- Though Linux supports long filenames that may contain embedded spaces and punctuation characters, limit the punctuation characters in the names of files you create to period, dash, and underscore. *Most important, do not embed spaces in filenames.* If you want to represent spaces between words in a filename, use underscore characters. You will thank yourself later.
- Linux has no concept of a "file extension" like some other operating systems. You may name files any way you like. The contents or purpose of a file is determined by other means. Although Unix-like operating systems don't use file extensions to determine the contents/purpose of files, many application programs do.

Some Helpful Shortcuts

Table 2-1 shows some useful ways to quickly change the current working directory.

Table 2-1: `cd` Shortcuts

Shortcut	Result
`cd`	Changes the working directory to your home directory.
`cd -`	Changes the working directory to the previous working directory.

Shortcut	Result
`cd`	Changes the working directory to the home directory of *`user_name`*. For example, typing `cd ~bob` will change the directory to the home directory of user “bob.”
`~user_name`	

Summing Up

This chapter explained how the shell treats the directory structure of the system. We learned about absolute and relative pathnames and the basic commands that we use to move around that structure. In the next chapter, we will use this knowledge to go on a tour of a modern Linux system.