CHAPTER 1

Starting with Linux

In This Chapter

Understanding Linux

Using Linux

Linux myths, legends, and FUD

In your hands, you have a dozen Linux distributions (on CD and DVD), thousands of applications, and descriptions to launch and get started with it all. For you right now, the worldwide Linux phenomenon is just a reboot away.

Linux is ready for prime time. Are you ready for Linux? Well, whether you know it or not, you probably run into Linux every day. When you buy a book from Amazon.com or search the Web with Google, you use Linux. You use Linux in your TiVo when you record TV shows and Linux may be running the PDA in your pocket. Animations you saw in the movie *Shrek 2* were created by hundreds of Linux workstations and rendered by a server farm of hundreds of other Linux systems.

Linux truly is everywhere.

Big computer companies, such as IBM, Oracle, Novell, and Red Hat, are lining up their products behind Linux. After dismissing it for years, companies such as Microsoft and Sun Microsystems are gathering their forces to deal with it. Who would have thought that some of the world's largest computer companies would fear a computer system built from code nobody can own that is given away for free?

But despite the fact that IBM featured Muhammad Ali in commercials for Linux during the Superbowl and that the mere mention of "Linux" for a dot-com company sent its stock through the roof in the 1990s, most people don't really know what Linux is. As Linux continues to improve exponentially, that's going to change.

Linux Bible 2006 Edition brings you into the world of free and open source software that, through some strange twists and turns, has fallen most publicly under the "Linux" banner. Through descriptions and procedures, this book helps you:

- ✦ Understand what Linux is and where it comes from
- ✦ Sort through the various distributions of Linux to choose one (or more) that is right for you (you get several on this book's CD and DVD)
- ✦ Try out Linux as a desktop computer, server computer, or programmer's workstation
- ✦ Become connected to the open source software movement, as well as many separate high-quality software projects that are included with Linux

Whether you are using Linux for the first time or just want to try out a new Linux distribution, *Linux Bible 2006 Edition* is your guide to using Linux and the latest open source technology. While different Linux distributions vary in the exact software they include, this book describes the most popular software available for Linux to:

- ✦ Manage your desktop (menus, icons, windows, and so on)
- ✦ Listen to music and watch video
- ✦ Use word processor, spreadsheet, and other office productivity applications
- ✦ Browse the Web and send e-mail
- ✦ Play games
- ✦ Find thousands of other open source software packages you can get for free

Because most Linux distributions also include features that let them act as servers (in fact, that's what Linux has always been best at), you'll also learn about software available for Linux that lets you do the following:

- ✦ Connect to the Internet or other network
- ✦ Use Linux as a firewall, router, and DHCP server to protect and manage your private network
- ✦ Run a Web server (using Apache, MySQL, and PHP)
- ✦ Run a mail server (using exim or other mail transfer agent)
- ✦ Run a print server (using Samba or CUPS)
- ✦ Run a file server (using FTP or Samba)

This book guides you through the basics of getting started with the Linux features just mentioned, plus many more features that I'll get to later. You'll go through the following basic steps:

1. **Understanding Linux.** You need to know where Linux came from, how it is developed, and how it's ultimately packaged. This chapter describes the UNIX heritage on which Linux was founded, the free and open source software development efforts underway, and the organizations and individuals that package and produce Linux distributions.

2. **Trying Linux.** In the past, an impediment to trying Linux was getting it installed on a computer that was devoted solely to Microsoft Windows. With bootable Linux systems such as KNOPPIX (and others included with this book), you can boot a fully functioning Linux from DVD, CD, or floppy disk without disturbing the current contents of your computer.
3. **Installing Linux.** You can install a fully functioning Linux system permanently on your hard disk. Disk space required varies from a few hundred megabytes for a minimal installation to 6 gigabytes for a full range of desktop, server, and programming features. Chapters in Part III, "Choosing and Installing a Linux Distribution," describe how to install several different Linux distributions.
4. **Using Linux.** You won't know if Linux can be used to replace your current desktop or server system until you start using it. This book helps you try OpenOffice.org software to write documents, create spreadsheets, and build presentations. It describes xmms and mplayer for playing your music and video content, respectively, and covers some of the best Linux tools available for Web browsing (for example, Firefox, Mozilla and Konqueror) and managing your e-mail (such as Evolution and Thunderbird).
5. **Configuring Linux.** Linux works very well as a desktop system, and it can also be configured to act as a router, a firewall, and a variety of server types. While there are some excellent graphical tools for administering Linux systems, most Linux administrators edit configuration files and run commands to configure Linux. Part II, "Running the Show," contains basic information for administering Linux, and Part V, "Running Servers," discusses procedures for setting up various types of servers.

Once you've been through the book, you should be proficient enough to track down your more advanced questions through the volumes of man pages, FAQs, HOW-TOs, and forums that cover different aspects of the Linux operating system.

Understanding Linux

People who don't know what Linux is sometimes ask me if it's a program that runs on Microsoft Windows. When I tell them that Linux is, itself, an operating system like Windows and that they can remove (or never purchase) Windows, I sometimes get a surprised reaction: "A PC can run with nothing from Microsoft on it?"

Yes, Linux is a full-blown operating system that is a free clone of the UNIX operating system. Start your computer with Linux, and Linux takes over the operation of your PC and manages the following aspects of your computer:

- **Processor —** Because Linux can run many processes from many different users at the same time (even with multiple CPUs on the same machine), Linux needs to be able to manage those processes. The Linux scheduler sets the priorities for running tasks and manages which processes run on which CPUs (if multiple processors are present). The scheduler can be tuned differently for

different types of Linux systems. If it's tuned properly, the most important processes get the quickest responses from the processor. For example, a Linux scheduler on a desktop system gives higher priority to things such as moving a window on the desktop than it does to a background file transfer.

- **Memory** — Linux tries to keep processes with the most immediate need in RAM, while managing how processes that exceed the available memory are moved to swap space. *Swap space* is a defined area on your hard disk that's used to handle the overflow of running processes and data. When RAM is full, processes are placed in swap space. When swap space is full (something that you don't want to happen), new processes can't start up.
- **Devices** — Linux supports thousands of hardware devices, yet keeps the kernel a manageable size by including only a small set of drivers in the active kernel. Using loadable modules, the kernel can add support for other hardware as needed. Modules can be loaded and unloaded on demand, as hardware is added and removed. (The kernel, described in detail a bit later on, is the heart of the Linux operating system.)
- **File systems** — File systems provide the structure in which files are stored on hard disk, CD, DVD, floppy disks, or other media. Linux knows about different file system types (such as Linux ext3 and reiserfs file systems, or VFAT and NTFS from Windows systems) and how to manage them.
- **Security** — Like UNIX, Linux was built from the ground up to enable multiple users to access the system simultaneously. To protect each user's resources, every file, directory, and application is assigned sets of read, write, and execute permissions that define who can access them. In a standard Linux system, the root user has access to the entire system, some special logins have access to control particular services (such as Apache for Web services), and users can be assigned permission individually or in groups. Recent features such as Security-Enhanced Linux enable more refined tuning and protection in highly secure computing environments.

What I have just described are components that primarily make up what is referred to as the Linux *kernel*. In fact, the Linux kernel (which was created and is still managed by Linus Torvalds) is what gives Linux its name. The kernel is the software that starts up when you boot your computer and manages the programs you use so they can communicate effectively and simply with your computer hardware.

Other components, such as administrative commands and applications, are added to the kernel from other free and open source software projects to make Linux a complete operating system. The GNU project, in particular, contributed many components that are now in Linux. (GNU, Apache, KDE, GNOME, and other key open source projects in Linux are discussed a bit later). Those other projects added such things as:

- **Graphical user interfaces (GUIs)** — Consisting of a graphical framework (typically the X Window System), window managers, panels, icons, and menus. GUIs enable you to use Linux with a keyboard and mouse combination, instead of just typing commands (as was done in the old days).

- **Administrative utilities** — Including hundreds (perhaps thousands) of commands and graphical windows to do such things as add users, manage disks, monitor the network, install software, and generally secure and manage your computer.
- **Applications** — Although no Linux distribution includes all of them, there are literally thousands of games, office productivity tools, Web browsers, chat windows, multimedia players, and other applications available for Linux.
- **Programming tools** — Including programming utilities for creating applications and libraries for implementing specialty interfaces.
- **Server features** — Enabling you to offer services from your Linux computer to another computer on the network. In other words, while Linux includes Web browsers to view Web pages, it can also be the computer that serves up Web pages to others. Popular server features include Web, mail, database, printer, file, DNS, and DHCP servers.

Once Linus Torvalds and friends had a working Linux kernel, pulling together a complete open source operating system was possible because so much of the available "free" software was:

- Covered by the GNU Public License (GPL) or similar license. That allowed the entire operating system to be freely distributed, provided guidelines were followed relating to how the source code for that software was made available going forward (see `http://www.gnu.org/licenses/gpl.html`).
- Based on UNIX-like systems. Clones of virtually all the other user-level components of a UNIX system had been created. Those and other utilities and applications were built to run on UNIX or other UNIX-like systems.

Linux has become the culmination of the open source software movement. But the traditions of sharing code and building communities that made Linux possible started years before Linux was born. You could argue that it began in a comfortable think tank known as Bell Laboratories.

Exploring Linux History

Some histories of Linux begin with this message posted by Linus Torvalds to the `comp.os.minix` newsgroup on August 25, 1991:

> Hello everybody out there using minix -
>
> I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones. This has been brewing since april, and is starting to get ready. I'd like any feedback on things people like/dislike in minix, as my OS resembles it somewhat (same physical layout of the file-system (due to practical reasons) among other things) . . . Any suggestions are welcome, but I won't promise I'll implement them :-)

> Linus (torvalds@kruuna.helsinki.fi)
>
> PS. Yes — it's free of any minix code, and it has a multi-threaded fs. It is NOT protable[sic] (uses 386 task switching etc), and it probably never will support anything other than AT-harddisks, as that's all I have :-(.
>
> Reprinted from Linux International Web site (www.li.org/linuxhistory.php)

Minix was a UNIX-like operating system that ran on PCs in the early 1990s. Like Minix, Linux was also a clone of the UNIX operating system. To truly appreciate how a free operating system could have been modeled after a proprietary system from AT&T Bell Laboratories, it helps to understand the culture in which UNIX was created and the chain of events that made the essence of UNIX possible to reproduce freely.

From a Free-Flowing UNIX Culture at Bell Labs

From the very beginning, the UNIX operating system was created and nurtured in a communal environment. Its creation was not driven by market needs, but by a desire to overcome impediments to producing programs. AT&T, which owned the UNIX trademark originally, eventually made UNIX into a commercial product, but by that time, many of the concepts (and even much of the early code) which made UNIX special had fallen into the public domain.

If you are under 30 years old, you may not remember a time when AT&T was "the" phone company. Up until the early 1980s, AT&T didn't have to think much about competition because if you wanted a phone in the United States, you had to go to AT&T. It had the luxury of funding pure research projects. The Mecca for such projects was the Bell Laboratories site in Murray Hill, New Jersey.

After the failure of a project called Multics in around 1969, Bell Labs employees Ken Thompson and Dennis Ritchie set off on their own to create an operating system that would offer an improved environment for developing software. Up to that time, most programs were written on punch cards that had to be fed in batches to mainframe computers. In a 1980 lecture on "The Evolution of the UNIX Time-sharing System," Dennis Ritchie summed up the spirit that started UNIX:

> What we wanted to preserve was not just a good environment in which to do programming, but a system around which a fellowship could form. We knew from experience that the essence of communal computing as supplied by remote-access, time-shared machines is not just to type programs into a terminal instead of a keypunch, but to encourage close communication.

The simplicity and power of the UNIX design began breaking down barriers that impeded software developers. The foundation of UNIX was set with several key elements:

- **The UNIX file system** — After creating the structure that allowed levels of subdirectories (which, for today's desktop users, looks like folders inside of folders), UNIX could be used to organize the files and directories in intuitive ways. Furthermore, complex methods of accessing disks, tapes, and other devices were greatly simplified by representing those devices as individual device files that you could also access as items in a directory.
- **Input/output redirection** — Early UNIX systems also included input redirection and pipes. From a command line, UNIX users could direct the output of a command to a file using a right arrow key (`>`). Later, the concept of pipes was added (`|`) where the output of one command could be directed to the input of another command. For example, the command line

  ```
  $ cat file1 file2 | sort | pr | lpr
  ```

  concatenates (`cat`) file1 and file2, sorts (`sort`) the lines in those files alphabetically, paginates the sorted text for printing (`pr`), and directs the output to the computer's default printer (`lpr`). This method of directing input and output enabled developers to create their own specialized utilities that could be joined together with existing utilities. This modularity made it possible for lots of code to be developed by lots of different people.
- **Portability** — Much of the early work in simplifying the experience of using UNIX led to its also becoming extraordinarily portable to run on different computers. By having device drivers (represented by files in the file system tree), UNIX could present an interface to applications in such a way that the programs didn't have to know about the details of the underlying hardware. To later port UNIX to another system, developers had only to change the drivers. The applications program didn't have to change for different hardware!

  To make the concept of portability a reality, however, a high-level programming language was needed to implement the software needed. To that end, Brian Kernighan and Dennis Ritchie created the C programming language. In 1973, UNIX was rewritten in C. Today, C is still the primary language used to create the UNIX (and Linux) operating system kernels.

As Ritchie went on to say in his 1980 lecture:

> Today, the only important UNIX program still written in assembler is the assembler itself; virtually all the utility programs are in C, and so are most of the applications programs, although there are sites with many in Fortran, Pascal, and Algol 68 as well. It seems certain that much of the success of UNIX follows from the readability, modifiability, and portability of its software that in turn follows from its expression in high-level languages.

If you are a Linux enthusiast and are interested in what features from the early days of Linux have survived, an interesting read is Dennis Ritchie's reprint of the first UNIX programmer's manual (dated November 3, 1971). You can find it at Dennis Ritchie's Web site: `http://cm.bell-labs.com/cm/cs/who/dmr/1stEdman.html`.

The form of this documentation is UNIX man pages — which is still the primary format for documenting UNIX and Linux operating system commands and programming tools today.

What's clear as you read through the early documentation and accounts of the UNIX system is that the development was a free-flowing process, lacked ego, and was dedicated to making UNIX excellent. This process led to a sharing of code (both inside and outside of Bell Labs) that allowed rapid development of a high-quality UNIX operating system. It also led to an operating system that AT&T would find difficult to reel back in later.

To a Commercialized UNIX

Before AT&T divestiture in 1984, when it was split up into AT&T and seven "baby Bell" companies, AT&T was forbidden to sell computer systems. Companies you now know by names such as Verizon, Qwest, SBC Communications, and Lucent Technologies were all part of AT&T. As a result of AT&T's monopoly of the telephone system, the U.S. government was concerned that an unrestricted AT&T might dominate the fledgling computer industry.

Because AT&T was restricted from selling computers directly to customers before its divestiture, UNIX source code was licensed to universities for a nominal fee. There was no UNIX operating system for sale from AT&T that you didn't have to compile yourself.

BSD Arrives

In 1975, UNIX V6 became the first version of UNIX available for widespread use outside of Bell Laboratories. From this early UNIX source code, the first major variant of UNIX was created at University of California at Berkeley. It was named the Berkeley Software Distribution (BSD).

For most of the next decade, the BSD and Bell Labs versions of UNIX headed off in separate directions. BSD continued forward in the free-flowing, share-the-code manner that was the hallmark of the early Bell Labs UNIX, while AT&T started steering UNIX toward commercialization. With the formation of a separate UNIX Laboratory, which moved out of Murray Hill and down the road to Summit, New Jersey, AT&T began its attempts to commercialize UNIX. By 1984, divestiture was behind AT&T and it was ready to really start selling UNIX.

UNIX Laboratory and Commercialization

The UNIX Laboratory was considered a jewel that couldn't quite find a home or a way to make a profit. As it moved between Bell Laboratories and other areas of AT&T, its name changed several times. It is probably best remembered by its last name, which it had as it began its spin-off from AT&T: UNIX System Laboratories (USL).

The UNIX source code that came out of USL, the legacy of which is now owned by Santa Cruz Operation (SCO), is being used as the basis for lawsuits by SCO against major Linux vendors (such as IBM and Red Hat, Inc.). Because of that, I think the efforts from USL that have contributed to the success of Linux are sometimes disrespected.

You have to remember that, during the 1980s, many computer companies were afraid that a newly divested AT&T would pose more of a threat to controlling the computer industry than would an upstart company in Redmond, Washington. To calm the fears of IBM, Intel, DEC, and others computer companies, the UNIX Lab made the following commitments to ensure a level playing field:

- **Source code only** — Instead of producing their own boxed set of UNIX, AT&T continued to sell only source code and to make it available equally to all licensees. Each company would then port UNIX to its own equipment. It wasn't until about 1992, when the lab was spun off as a joint venture with Novell (called Univel), and then eventually sold to Novell, that a commercial boxed set of UNIX (called UnixWare) was produced directly from that source code.
- **Published interfaces** — To create an environment of fairness and community to its OEMs (original equipment manufacturers), AT&T began standardizing what different ports of UNIX had to be able to do to still be called UNIX. To that end, Portable Operating System Interface (POSIX) standards and the AT&T UNIX System V Interface Definition (SVID) were specifications UNIX vendors could use to create compliant UNIX systems. Those same documents also served as road maps for the creation of Linux.

In an early e-mail newsgroup post, Linus Torvalds made a request for a copy, preferably online, of the POSIX standard. I think that nobody from AT&T expected someone to actually be able to write their own clone of UNIX from those interfaces, without using any of its UNIX source code.

- **Technical approach** — Again, until the very end of USL, most decisions on the direction of UNIX were made based on technical considerations. Management was promoted up through the technical ranks and to my knowledge there was never any talk of writing software to break other companies' software or otherwise restrict the success of USL's partners.

When USL eventually started taking on marketing experts and creating a desktop UNIX product for end users, Microsoft Windows already had a firm grasp on the desktop market. Also, because the direction of UNIX had always been toward source-code licensing destined for large computing systems, USL had pricing difficulties for its products. For example, on software it was including with UNIX, USL found itself having to pay out per-computer licensing fees that were based on $100,000 mainframes instead of $2,000 PCs. Add to that the fact that no application programs were available with UnixWare, and you can see why the endeavor failed.

Successful marketing of UNIX systems at the time, however, was happening with other computer companies. SCO had found a niche market, primarily selling PC versions of UNIX running dumb terminals in small offices. Sun Microsystems was selling lots of UNIX workstations (originally based on BSD but merged with UNIX in SVR4) for programmers and high-end technology applications (such as stock trading).

Other commercial UNIXes were also emerging by the 1980s as well. This new ownership assertion of UNIX was beginning to take its toll on the spirit of open contributions. Lawsuits were being raised to protect UNIX source code and trademarks. In 1984, this new, restrictive UNIX gave rise to an organization that eventually led a path to Linux: the Free Software Foundation.

To a GNU Free-Flowing (not) UNIX

In 1984, Richard M. Stallman started the GNU project (`www.gnu.org`), recursively named by the phrase GNU is Not UNIX. As a project of the Free Software Foundation (FSF), GNU was intended to become a recoding of the entire UNIX operating system that could be freely distributed.

While rewriting millions of lines of code might seem daunting to one or two people, spreading the effort across dozens or even hundreds of programmers made the project possible. It turned out that not only could the same results be gained by all new code, but that in some cases that code was better than the original UNIX versions. Because everyone could see the code being produced for the project, poorly written code could be corrected quickly or replaced over time.

If you are familiar with UNIX, try searching the more than 3,400 GNU software packages for your favorite UNIX command from the Free Software Directory (`http://directory.fsf.org/GNU`). Chances are you will find it there, along with many, many other software projects available as add-ons.

Over time, the term *free software* has been mostly replaced by the term *open source software*. As a nod to both the two camps, however, some people use the term Free and Open Source Software (FOSS) instead. An underlying principal of FOSS, however, is that, while you are free to use the software as you like, you have some responsibility to make the improvements you make to the code available to others. In that way, everyone in the community can benefit from your work as you have benefited from the work of others'.

To clearly define how open source software should be handled, the GNU software project created the GNU Public License (you can read the GPL in its entirety at the end of this book). While there are many other software licenses covering slightly different approaches to protecting free software, the GPL is perhaps the most well known — and it's the one that covers the Linux kernel itself. Basic features of the GNU Public License include:

- ✦ **Author rights** — The original author retains the rights to his or her software.
- ✦ **Free distribution** — People can use the GNU software in their own software, changing and redistributing it as they please. They do, however, have to include the source code with their distribution (or make it easily available).
- ✦ **Copyright maintained** — Even if you were to repackage and resell the software, the original GNU agreement must be maintained with the software, which means all future recipients of the software have the opportunity to change the source code, just as you did.

There is no warranty on GNU software. If something goes wrong, the original developer of the software has no obligation to fix the problem. However, there are many organizations, big and small, that offer paid support packages for the software when it is included in their Linux or other open source software distribution. (See the OSI Open Source Definition later in this chapter for a more detailed definition of open source software.)

Despite its success producing thousands of UNIX utilities, the GNU project itself failed to produce one critical piece of code: the kernel. Its attempts to build an open source kernel with the GNU Hurd project (`www.gnu.org/software/hurd`) were unsuccessful.

BSD Loses Some Steam

The one software project that had a chance of beating out Linux to be the premier open source software project was the venerable old BSD project. By the late 1980s, BSD developers at UC Berkeley realized that they had already rewritten most of the UNIX source code they had received a decade earlier.

In 1989, UC Berkeley distributed its own UNIX-like code as Net/1 and later (in 1991) as Net/2. Just as UC Berkeley was preparing a complete, UNIX-like operating system that was free from all AT&T code, AT&T hit them with a lawsuit in 1992. The suit claimed that the software was written using trade secrets taken from AT&T's UNIX system.

The lawsuit was dropped when Novell bought UNIX System Laboratories from AT&T in 1994. But, during that critical time period, there was enough fear and doubt about the legality of the BSD code that the momentum BSD had gained to that point in the fledgling open source community was lost. Many people started looking for another open source alternative. The time was ripe for a college student from Finland who was working on his own kernel.

Today, BSD versions are available from three projects: FreeBSD, NetBSD, and OpenBSD. People generally characterize FreeBSD as the easiest to use, NetBSD as available on the most computer hardware platforms, and OpenBSD as fanatically secure. Many security-minded individuals still prefer BSD over Linux.

Linus Builds the Missing Piece

In 1991, Linus Torvalds, a student at the University of Helsinki, Finland, started work on a UNIX-like kernel because he wanted to be able to use the same kind of operating system on his home PC that he used at school. At the time, Linus was using Minix, but he wanted to go beyond what the Minix standards permitted.

As noted earlier, Linus announced the first public version of the Linux kernel to the comp.os.minix newsgroup on August 25, 1991, although Linus guesses that the first version didn't actually come out until mid-September of that year (see the Linux International Web site's Linux History page: `www.li.org/linuxhistory.php`).

Although Torvalds stated that Linux was written for the 386 processor and probably wasn't portable, others persisted in encouraging (and contributing to) a more portable approach in the early versions of Linux. By October 5, Linux 0.02 was released with much of the original assembly code rewritten in the C programming language, which made it possible to start porting it to other machines.

The Linux kernel was the last — and the most important — piece of code that was needed to complete a whole UNIX-like operating system under the GPL. So, when people started putting together distributions, the name Linux and not GNU is what stuck. Some distributions such as Debian, however, refer to themselves as GNU/Linux distributions.

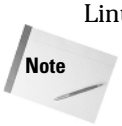
Within the next few years, commercial and non-commercial Linux distributions began to emerge. MCC Interim Linux (`ftp.mcc.ac.uk/pub/linux/distributions/MCC`) was released in the U.K. in February, 1992. Slackware Linux (described in Chapter 14), which was first released in April, 1993, is one of the oldest surviving Linux distributions.

Today, Linux can be described as an open source UNIX-like operating system that reflects a combination of SVID, POSIX, and BSD compliance. Linux continues to aim toward compliance with POSIX as well as with standards set by the new owner of the UNIX trademark, The Open Group (`www.unix-systems.org`).

The non-profit Open Source Development Labs (`www.osdl.org`), which employs Linus Torvalds, manages the direction today of Linux development efforts. Its sponsors' list is like a Who's Who of commercial Linux vendors, including IBM, Red Hat, SUSE (Novell), VA Software, HP, Dell, Computer Associates, Intel, Cisco Systems, and others. OSDL's primary charter is to accelerate the growth of Linux in telecommunications and data centers.

Although much of the thrust of corporate Linux efforts is on corporate, enterprise computing, huge improvements are continuing in the desktop arena as well. The KDE and GNOME desktop environments continuously improve the Linux experience for casual users. Major efforts are underway to offer critical pieces of desktop components that are still not available in open source versions, including multimedia software and office productivity applications.

Linus continues to maintain and improve the Linux kernel.



To get more detailed histories of Linux, I recommend visiting the LWN.net site. LWN.net has kept a detailed Linux timeline from 1998 to the present day. For example, the 2003 timeline is available at `http://lwn.net/Articles/Timeline2003`. Another good resource is the book *Open Sources: Voices from the Open Source Revolution* (O'Reilly). The whole first edition (published in 1999) is available online (`www.oreilly.com/catalog/opensources/book/toc.html`).

What's So Great About Linux?

Leveraging work done on UNIX and GNU projects helped to get Linux up and running quickly. The culture of sharing in the open source community and adoption of a wide array of tools for communicating on the Internet have helped Linux to move quickly through infancy and adolescence to become a mature operating system.

The simple commitment to share code is probably the single most powerful contributor to the growth of the open source software movement in general, and Linux in particular. That commitment has also encouraged involvement from the kind of people who are willing to contribute back to that community in all kinds of ways. The willingness of Linus to incorporate code from others in the Linux kernel has also been critical to the success of Linux.

The following sections characterize Linux and the communities that support it.

Features in Linux

If you have not used Linux before, you should expect a few things to be different from using other operating systems. Here is a brief list of some Linux features that you might find cool:

- **No constant rebooting** — Uptime is valued as a matter of pride (remember, Linux and other UNIX systems are most often used as servers, which are expected to stay up 24/7). After the original installation, you can install or remove most software without having to reboot your computer.
- **Start/stop services without interrupting others** — You can start and stop individual services (such as Web, file, and e-mail services) without rebooting or even interrupting the work of any other users or features of the computer. In other words, you should not have to reboot your computer every time someone sneezes.
- **Portable software** — You can usually change to another Linux, UNIX, or BSD system and still use the exact same software! Most open source software projects were created to run on any UNIX-like system and many also run on Windows systems, if you need them to. If it won't run where you want it to,

chances are that you, or someone you hire, can port it to the computer you want. (*Porting* refers to modifying an application or driver so it works in a different computer architecture or operating system.)

- **Downloadable applications** — If the applications you want are not delivered with your version of Linux, you can often download and install them with a single command, using tools such as apt and yum.
- **No settings hidden in code or registries** — Once you learn your way around Linux, you'll find that (given the right permissions on your computer) most configuration is done in plain text files that are easy to find and change.
- **Mature desktop** — The X Window System (providing the framework for your Linux desktop) has been around longer than Microsoft Windows. The KDE and GNOME desktop environments provide graphical interfaces (windows, menus, icons, and so forth) that rival those on Microsoft systems. Ease-of-use problems with Linux systems are rapidly evaporating.
- **Freedom** — Linux, in its most basic form, has no corporate agenda or bottom line to meet. You are free to choose the Linux distribution that suits you, look at the code that runs the system, add and remove any software you like, and make your computer do what you want it to do.

There are some aspects of Linux that make it hard for some new users to get started. One is that Linux is typically set up to be secure by default, so you need to adjust to using an administrative login (root) to make most changes that affect the whole computer system. Although this can be a bit inconvenient, trust me, it makes your computer safer than just letting anyone do anything.

For the same reason, many services are off by default, so you need to turn them on and do at least minimal configuration to get them going. For someone who is used to Windows, Linux can be difficult just because it is different than Windows. But because you're reading this book, I assume you want to learn about those differences.

OSI Open Source Definition

For software developers, Linux provides a platform that lets them change the operating system as they like and get a wide range of help creating the applications they need. One of the watchdogs of the open source movement is the Open Source Initiative (`www.opensource.org`). This is how the OSI Web site describes open source software:

> The basic idea behind open source is very simple: When programmers can read, redistribute, and modify the source code for a piece of software, the software evolves. People improve it, people adapt it, people fix bugs. And this can happen at a speed that, if one is used to the slow pace of conventional software development, seems astonishing.

> We in the open source community have learned that this rapid evolutionary process produces better software than the traditional closed model, in which only a very few programmers can see the source and everybody else must blindly use an opaque block of bits.

While the primary goal of open source software is to make source code available, other goals are also defined by OSI in its Open Source Definition. Most of the following rules for acceptable open source licenses are to protect the freedom and integrity of the open source code:

- **Free distribution** — An open source license can't require a fee from anyone who resells the software.
- **Source code** — The source code has to be included with the software and not be restricted from being redistributed.
- **Derived works** — The license must allow modification and redistribution of the code under the same terms.
- **Integrity of the author's source code** — The license may require that those who use the source code remove the original project's name or version if they change the source code.
- **No discrimination against persons or groups** — The license must allow all people to be equally eligible to use the source code.
- **No discrimination against fields of endeavor** — The license can't restrict a project from using the source code because it is commercial or because it is associated with a field of endeavor that the software provider doesn't like.
- **Distribution of license** — No additional license should be needed to use and redistribute the software.
- **License must not be specific to a product** — The license can't restrict the source code to a particular software distribution.
- **License must not restrict other software** — The license can't prevent someone from including the open source software on the same medium as non–open source software.
- **License must be technology-neutral** — The license can't restrict methods in which the source code can be redistributed.

Open source licenses used by software development projects must meet these criteria to be accepted as open source software by OSI. More than 40 different licenses are accepted by OSI to be used to label software as "OSI Certified Open Source Software." In addition to the GPL, other popular OSI-approved licenses include:

- **LGPL** — The GNU Lesser General Public License (LGPL) allows people to redistribute certain software, but not change its contents. This license is often used for distributing libraries that other application programs depend upon.

- **BSD** — The Berkeley Software Distribution License allows redistribution of source code, with the requirement that the source code keep the BSD copyright notice and not use the names of contributors to endorse or promote derived software without written permission.
- **MIT** — The MIT license is like the BSD license, except that it doesn't include the endorsement and promotion requirement.
- **Mozilla** — The Mozilla license covers use and redistribution of source code associated with the Mozilla Web browser and related software. It is a much longer license than the others just mentioned because it contains more definitions of how contributors and those reusing the source code should behave. This includes submitting a file of changes when submitting modifications and that those making their own additions to the code for redistribution should be aware of patent issues or other restrictions associated with their code.

The end result of open source code is software that has more flexibility to grow and fewer boundaries in how it can be used. Many believe that the fact that many people look over the source code for a project will result in higher quality software for everyone. As open source advocate Eric S. Raymond says in an often-quoted line, "Many eyes make all bugs shallow."

Vibrant Communities

Communities of professionals and enthusiasts have grown around Linux and its related open source projects. Many have shown themselves willing to devote their time, knowledge, and skills on public mailing lists, forums, Wikis, and other Internet venues (provided you ask politely and aren't too annoying).

Linux User Groups (LUGs) have sprung up all over the world. Many LUGs sponsor Linux installfests (where members help you install the Linux of your choice on your computer) or help non-profit groups and schools use Linux on older computers that will no longer support the latest Microsoft Windows software. The LUG I'm a member of holds monthly meetings with talks on Linux topics and has an active Web site, mailing list, and chat server where members can help each other with Linux questions that come up.

Free online bulletin board services have sprung up to get information on specific Linux topics. Popular general Linux forums are available from `www.LinuxQuestions.org`, `www.LinuxForums.org`, and `www.LinuxHelp.net`. Most of these sites are built with open source software (see `www.e107.org` and `www.phpBB.com` for examples of open source forum software).

Communities also gather around specific software projects and Linux distributions. `www.sourceforge.net` is the home to thousands of open source software projects. Go to the SourceForge.net site and try keyword searches for topics that interest you (for example, image gallery or video editing). Each project provides links to project home pages, forums, and software download sites. There are always projects looking for people to help write code or documentation or just participate in discussions.

You'll find that most major Linux distributions have associated mailing lists and forums. You can go directly to the Web sites for Red Hat Fedora Linux (`www.redhat.com/fedora`), Debian (`www.debian.com`), SUSE (`www.novell.com/linux/suse`), Gentoo (`www.gentoo.org`), and others to learn how to participate in forums and contribute to those projects.

Major Software Projects

Some software projects have grown beyond the status of being simply a component of Linux or some other UNIX derivative. Some of these projects are sponsored and maintained by organizations that oversee multiple open source projects. This section introduces some of the most popular open source projects and organizations.

- **The Apache Software Foundation** (`www.apache.org`) is not only the world's most popular open source Web server software, it's the most popular of all Web server software. Most Linux distributions that contain server software include Apache. The Apache Software Foundation maintains the Apache Web (HTTP) server and about a dozen other projects, including SpamAssassin (for blocking and filtering e-mail spam), Apache Portals (to provide portal software), and a bunch of projects for producing modules to use with your Apache Web server.
- **The Internet Systems Consortium** (`www.isc.org`) supports critical Internet infrastructure projects under open-source licenses. Those projects include Bind (DNS server software), DHCP (to assign IP addresses and other information to Internet clients), INN (for creating Internet news servers) and OpenReg (a tool for managing delegation of domains in a shared registry).
- **The Free Software Foundation** (`www.fsf.org`) is the principal sponsor of the GNU Project. Most of the UNIX commands and utilities included in Linux that were not closely associated with the kernel were produced under the umbrella of the GNU project.
- **The Mozilla project's** (`www.mozilla.org`) most well-known product is the very popular open source Web browser Mozilla Navigator, which was originally based on code released to the open source community from Netscape Communicator, Most other open source browsers incorporate Mozilla's engine. The Mozilla project also offers related e-mail, composer, IRC Chat, and address book software. New projects include the Thunderbird e-mail and news client and Firefox Web browser, which have seen enormous success on Linux, Windows and Mac OSX platforms in the past year.
- **The Sendmail Consortium** (`www.sendmail.org`) maintains the sendmail mail transport agent, which is the world's most popular software for transporting mail across the Internet.

There are, of course, many more open source projects and organizations that provide software included in various Linux distributions, but the ones discussed here will give you a good feel for the kind of organizations that produce open source software.

Linux Myths, Legends, and FUD

The unlikely rise in the popularity of Linux has led to rampant (and sometimes strange) speculation about all the terrible things it could lead to or, conversely, to almost manic declarations of how Linux will solve all the problems of the world. I'll try as best I can (with my own admitted bias toward Linux) to present facts to address beliefs about Linux and to combat some of the unrealistic fear, uncertainty, and doubt (FUD) being spread by those with a vested interest in seeing Linux not succeed.

Can You Stop Worrying About Viruses?

Well, you can (and should) always worry about the security of any computer connected to the Internet. At the moment, however, you are probably less likely to get a virus from infected e-mail or untrusted Web sites with standard e-mail clients and Web browsers that come with Linux systems than you would with those that come with the average Microsoft Windows system.

The most commonly cited warnings to back up that statement come in a report from the United States Computer Emergency Readiness Team (CERT) regarding a vulnerability in Microsoft Internet Explorer (`www.kb.cert.org/vuls/id/713878`):

> There are a number of significant vulnerabilities in technologies relating to the IE domain/zone security model, the DHTML object model, MIME type determination, and ActiveX. It is possible to reduce exposure to these vulnerabilities by using a different Web browser, especially when browsing untrusted sites. Such a decision may, however, reduce the functionality of sites that require IE-specific features such as DHTML, VBScript, and ActiveX. Note that using a different Web browser will not remove IE from a Windows system, and other programs may invoke IE, the WebBrowser ActiveX control, or the HTML rendering engine (MSHTML).
>
> US-CERT Vulnerability Note VU#713878

While the note also recommends keeping up with patches from Microsoft to reduce your risks, it seems that the only real solutions are to disable Active scripting and ActiveX, use plain text e-mail, and don't visit sites you don't trust with Internet Explorer. In other words, use a browser that disables insecure features included in Microsoft products.

This announcement apparently caused quite a run on the Mozilla.org site to download a Mozilla or Firefox browser and related e-mail client (described in Chapter 22 of this book). Versions of those software projects run on Windows and Mac OS X, as well as on Linux. Many believe that browsers such as Mozilla are inherently more secure because they don't allow non-standard Web features that might do such things as automatically download unrequested software without your knowledge.

Of course, no matter what browser or e-mail client you are using, you need to follow good security practices (such as not opening attachments or downloading files you don't trust). Also, as open source browsers and e-mail clients, such as those from Mozilla.org, become more popular, the number of possible machines to infect through those applications will make it more tempting to virus writers. (At the moment, most viruses and worms are created specifically to attack Microsoft software.)

Will You Be Sued for Using Linux?

In the United States, anyone can try to sue anyone for anything. That doesn't mean that the people who bring lawsuits will win, but they can try. So far, the threat to individuals has not been substantial, but there have been some well-financed lawsuits against Linux providers. Those with litigation against Linux have gone primarily after big companies, such as IBM, Novell, and Red Hat, Inc., and have made only vague threats regarding end users of Linux. Linus Torvalds himself is the rare individual who has been named in lawsuits.

The SCO Lawsuits

The lawsuits getting the most press these days are the ones involving Santa Cruz Operation (SCO). SCO is the current owner of the UNIX source code that passed from AT&T Bell Labs to UNIX System Laboratories to Univel (a lot of people don't know that one) to Novell and eventually to the company formed by joining SCO and Caldera Systems.

Although the particulars of the claims seem to change daily, SCO's basic assertion in lawsuits against IBM and others is that Linux contains UNIX System V source code that is owned by SCO, so those who sell or use Linux owe licensing fees to SCO. To a layman (I am not a lawyer!), the assertions seem weak based on the facts that:

- ✦ There seems to be no original UNIX code in Linux. And, even if a small amount of code that could be proved to be owned by SCO had made it in there by mistake, that code could be easily dropped and rewritten.
- ✦ Concepts that created UNIX all seem to be in the public domain, with public specifications of UNIX interfaces blessed by AT&T itself in the form of published POSIX and System V Interface Definition standards.
- ✦ AT&T dropped a similar lawsuit in 1994 against BSD, which had actually started with UNIX source code, but rewritten it completely over the years.
- ✦ Exactly what SCO owns has been brought into question because Novell still claims some rights to the UNIX code it sold to SCO. (In fact, SCO doesn't even own the UNIX trademark, which Novell gave away to the Open Group before it sold the source code to SCO. Attempts were underway in 2004 by SCO to trademark the name UNIX System Laboratories.)

Responses to SCO's lawsuits (which certainly hold more weight than any explanations I could offer) are available from Open Group (`www.opengroup.org`), OSDL (`www.osdl.org`), IBM (`ibm.com/linux`), and Red Hat (`www.redhat.com`). The

Groklaw site (www.groklaw.net) is another good spot to learn about SCO lawsuits against Linux. If you are interested in the paper trail relating SCO's ownership of UNIX, I recommend the Novell's Unique Legal Rights page (www.novell.com/licensing/indemnity/legal.html).

OSDL.org has prepared a legal defense fund to protect Linux end users and other Linux litigants (including Linus and OSDL itself). You can read about this fund at ODSL's Linux Legal Defense Fund page (www.osdl.org/about_osdl/legal/lldf).

Software Patents

Few will argue that it is illegal for someone to copy a software company's code and redistribute it without permission. However, the concept of being able to patent an idea that a company might incorporate in its code has become a major point of contention in recent years. Can someone patent the idea of clicking an icon to open a window?

Software companies are scrambling to file thousands of patents related to how software is used. While those companies may never create products based on those patents, the restrictions those patents might place on other software companies or open source software development is a major issue.

The EuroLinux Alliance is a group dedicated to "protecting software freedom based on copyright, open standards, open competition, and open source software, such as Linux." EuroLinux is filing a petition in the European Parliament and European Council to warn about the dangers of software patents. To find out more, go to http://eurolinux.org.

Other Litigious Issues

Particularly contentious legal issues surround audio and video software. In Red Hat Linux 8, Red Hat, Inc. removed support for MP3 and DVD players because of questions about licensing associated with those music and movie formats. Red Hat's advice at the time was to download and install the players yourself for personal use. Red Hat didn't want to distribute those players because companies owning patents related to certain audio and video encoders might ask Red Hat to pay licensing fees for distributing those players (see www.redhat.com/advice/speaks_80mm.html).

Check with an attorney regarding any legal issues that concern you.

Can Linux Really Run on Everything from Handhelds to Supercomputers?

Linux is extraordinarily scalable and runs on everything from handhelds to supercomputers. Features in the Linux 2.6 kernel have been particularly aimed at making the kernel easier to port to embedded Linux systems, as well as large multi-processor, enterprise-quality servers.

Will Linux Crush Microsoft?

Linux doesn't seem to be trouncing Microsoft, although the rhetoric from Microsoft has targeted Linux as a particular threat in the server area. Microsoft is still the most popular desktop operating system in the world, holding more than 90 percent of the desktop market, by most accounts.

Major in-roads into the desktop market by Linux systems are expected to be slow in coming. However, an area where desktop Linux systems are making the greatest gains are in low-end, mass-market computers. For less than $300, you can buy a decent computer with Linspire Linux pre-installed from Wal-Mart, PC Clubs, or several other retailers. Because it is Linux, the system comes with a boat-load of applications as well (not Microsoft Office, but OpenOffice.org). (Linspire is discussed in Chapter 15.)

So far, most of the market share that Linux has gained has been taken from other UNIX systems, such as those from Sun Microsystems. Apache Web servers running on Linux are already considered the world's most popular Web server. With efforts underway from the likes of IBM, Oracle, Red Hat, and Novell, major pushes into the Enterprise market are already taking place. But Linux is still some distance from crushing Microsoft.

Are You on Your Own If You Use Linux?

If you are new to Linux and are concerned about support, there are several companies offering well-supported versions of Linux. Those include Red Hat Enterprise Linux (from Red Hat, Inc.) and SUSE Linux (from Novell, Inc.), as well as a number of other smaller players. In the corporate arena, add IBM to that list.

As noted earlier, there are also many community sites on the Internet that offer forums, mailing lists, and other venues for getting help if you get stuck.

Is Linux Only for Geeks?

It doesn't hurt to be a geek if you want to fully explore all the potential of your Linux system. However, with a good desktop Linux distribution, tremendous improvements over the past few years relating to ease-of-use and features have made it possible to do most things you would do on any Macintosh or Windows system without being a Linux expert.

Start with a Linux system that uses the KDE or GNOME desktop. Simple menus let you select word processors, Web browsers, games, and dozens of other applications you commonly use on other operating system. In most cases, you'll get along fine just using your mouse to work with windows, menus, and forms.

With Linux distributions that offer graphical tools for basic system administration (such as configuring a printer or network connection), you can be led through most tasks you need to do. Fedora, Red Hat Enterprise Linux, and SUSE are good examples

of Linux distributions that offer simplified administration tools. With a basic understanding of the Linux shell (see Chapter 2) and some help from a Linux forum, you should be able to troubleshoot most anything that goes wrong.

How Do Companies Make Money with Linux?

Open source enthusiasts believe that better software can result from an open source software development model than from proprietary development models. So in theory, any company creating software for its own use can save money by adding its software contributions to those of others to gain a much better end product for themselves.

Companies that want to make money selling software need to be more creative than they did in the old days. While you can sell the software you create that includes GPL software, you must pass the source code of that software forward. Of course, others can then recompile that product, basically using your product without charge. Here are a few ways that companies are dealing with that issue:

- **Software subscriptions** — Red Hat, Inc. sells its Red Hat Enterprise Linux products on a subscription basis. For a certain amount of money per year, you get binary code to run Linux (so you don't have to compile it yourself), guaranteed support, tools for tracking the hardware and software on your computer, and access to the company's knowledge base.

  While Red Hat's Fedora project includes much of the same software and is also available in binary form, there are no guarantees associated with the software or future updates of that software. A small office or personal user might take the risk on Fedora (which is itself an excellent operating system) but a big company that's running mission-critical applications will probably put down a few dollars for RHEL.

- **Donations** — Many open source projects accept donations from individuals or open source companies that use code from their projects. Amazingly, many open source projects support one or two developers and run exclusively on donations.

- **Bounties** — The concept of software bounties is a fascinating way for open source software companies to make money. Let's say that you are using XYZ software package and you need a new feature right away. By paying a software bounty to the project itself, or to other software developers, you can have your needed improvements moved to the head of the queue. The software you pay for will remain covered by its open source license, but you will have the features you need, at probably a fraction of the cost of building the project from scratch.

- **Boxed sets, mugs, and T-shirts** — Many open source projects have online stores where you can buy boxed sets (some people still like physical CDs and hard copies of documentation) and a variety of mugs, T-shirts, mouse pads, and other items. If you really love a project, for goodness sake, buy a T-shirt!

This is in no way an exhaustive list, because more creative ways are being invented every day to support those who create open source software. Remember that many people have become contributors to and maintainers of open source software because they needed or wanted the software themselves. The contributions they make for free are worth the return they get from others who do the same.

How Different Are Linux Distributions from One Another?

While different Linux systems will add different logos, choose some different software components to include, and have different ways of installing and configuring Linux, most people who become used to Linux can move pretty easily from one Linux to another. There are a few reasons for this:

- **Linux Standard Base** — There is an effort called the Linux Standard Base (`www.linuxbase.org`) to which most major Linux systems subscribe. The Linux Standard Base Specification (available from this site) has as one of its primary goals to ensure that applications written for one Linux system will work on other systems. To that end, the LSB will define what libraries need to be available, how software packages can be formatted, commands and utilities that must be available, and, to some extent, how the file system should be arranged. In other words, you can rely on many components of Linux being in the same place on LSB-certified Linux systems.
- **Open source projects** — Many Linux distributions include the same open source projects. So, for example, the most basic command and configuration files for an Apache Web server, Samba file/print server, and sendmail mail server will be the same whether you use Red Hat, Debian, or many other Linux systems. And although they can change backgrounds, colors, and other elements of your desktop, most of the ways of navigating a KDE or GNOME desktop stay the same, regardless of which Linux you use.
- **A shell is a shell** — Although you can put different pretty faces on it, once you open a shell command line interpreter (such as bash or sh) in Linux, most experienced Linux or UNIX users find it pretty easy to get around on most any Linux system. For that reason, I recommend that if you are serious about using Linux, you take some time to try the shell (as described in Chapter 2). Additionally, Chapters 24–27 focus on command line and configuration file interfaces for setting up servers because learning those ways of configuring servers will make your skills most portable across different Linux systems.

Is the Linux Mascot Really a Penguin?

Figure 1-1 shows the penguin logo that Linus Torvalds approved as the official Linux mascot. His name is Tux. Use of this logo is freely available, and you find it everywhere on Linux Web sites, magazines, and other Linux venues. (I used it in my book *Linux Toys II* and on the Linuxtoys.net Web site, for example.)

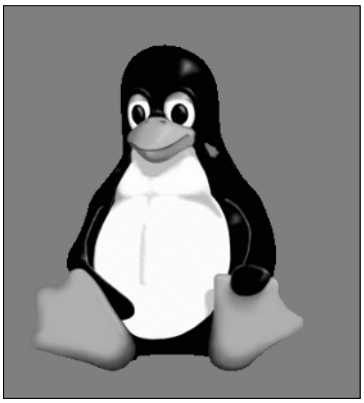


Figure 1-1: Tux, a gentle and pleasant penguin, is the official Linux mascot.

Tux was created by Larry Ewing. There are different versions of Tux available from his Web site (`www.isc.tamu.edu/~lewing/linux`). Find out more about Tux from the Linux Online Logos and Mascots page (`www.linux.org/info/logos.html`).

Getting Started with Linux

Although I've gone on a bit about Linux history and what it does, the primary goal of this book is to get you using it. To that end, I'd like to describe some things that might help you get started with Linux.

While Linux will run great on many low-end computers (even some old 486s and early Pentiums), if you are completely new to Linux, I recommend that you start with a PC that has a little more muscle. Here's why:

- ✦ Full-blown Linux operating systems with complete GNOME or KDE desktop environments perform poorly on slow CPUs and less than the recommended amount of RAM.
- ✦ You can create streamlined graphical Linux installations that will fit on small hard disks (as small as 100MB) and run fairly well on slow processors. However, putting together such a system requires some knowledge of which software packages to select and often requires some additional configuration.

If you are starting with a Pentium II, 400 MHz, your desktop will run slowly in default KDE or GNOME configurations with less than 128MB of RAM. A simpler desktop system, with just X and a window manager, will work, but won't give you the full flavor of a Linux desktop. (See Chapter 3 for information about different desktop choices and features.)

The good news is that, as mentioned earlier, cheap computers that you can buy from Wal-Mart or other retailers start at less than $300. Those systems will perform

better than most PCs you have laying around that are more than a few years old and will come with Linux (usually Linspire) pre-installed. The bottom line is that the less you know about Linux, the more you should try to have computer hardware that is up to spec to have a pleasant experience.

If you already have a Linux system sitting in front of you, Chapters 2 through 6 will walk you through the Linux shell, using the desktop, and some basic system administration. If you don't have a Linux system running on your computer yet, you have a couple of choices:

- **Try a bootable Linux** — If you have another OS on your machine and are reluctant to disturb the contents of your computer, a bootable Linux enables you to run Linux directly from a removable medium (DVD, CD, or even a floppy disk in some cases). You'll be able to try Linux without even touching the contents of your hard disk.
- **Install Linux on your hard disk** — If you have available disk space that's not already assigned to Windows or another system, you can install Linux on your hard disk and have a more permanent operating system. (Some Linux distributions, such as SUSE and Mandriva, let you resize your Windows hard disk to make room to install Linux.)

Linux itself is just a kernel (like the engine of a car), so to use Linux you need to select a Linux distribution. Because the distribution you choose is so critical to your Linux experience, the entire Part III of this book is devoted to understanding, choosing, and installing the most popular Linux distributions. Several of these distributions are included with this book, along with several useful bootable Linux distributions. If you don't already have a Linux system in front of you, refer to Chapter 7 to get started getting the Linux you want.

Summary

Linux is the most popular representation of the open source software model today and reflects a rich history of shared software development techniques that date back to the first UNIX systems of three decades ago. Today's Linux computer systems form the backbone of many major computing centers around the world.

In recent years, Linux has become a great choice as a desktop system as well. You will find many open source applications available for any type of application you can imagine (word processing, music playing, e-mail, games, and so on). With its powerful networking and built-in security features, Linux can provide a much safer computing environment than other desktop computing systems.

Linux gives you the freedom to create the kind of computer system you need.

✦ ✦ ✦