Preface

The Unix Family of Operating Systems

An *operating system* (or "OS") is a set of programs that controls a computer. It controls both *hardware* (things you can touch, like keyboards, screens, and disk drives) and *software* (application programs that you run, like a word processor).

Some computers have a *single-user* OS, which means that only one person can use the computer at a time. Many older OSes (such as MS-DOS) can also do only one job at a time. But almost any computer can do a lot more if it has a *multiuser, multitasking* operating system such as Unix. These powerful OSes let many people use the computer at the same time and let each user run several jobs at once.

Unix was invented more than 30 years ago for scientific and professional users who wanted a very powerful and flexible OS. It's been significantly developed since then. Because Unix was designed for experts, it can be a bit overwhelming at first. But after you get the basics (from this book!) you'll start to appreciate some of the reasons to use Unix:

- It comes with a huge number of powerful application programs. You can get many others for free on the Internet. (The GNU utilities, available from the Free Software Foundation, are very popular.) You can thus do much more at a much lower cost.
- Not only are the applications often free, but some Unix versions are also free. Linux is a good example. Like the free applications, most free Unix versions are of excellent quality. They're maintained by

volunteer programmers who want a powerful OS and are frustrated by the slow, bug-ridden OS development at some big software companies.

- Unlike OSes such as Microsoft Windows and MacOS that are designed for certain types of hardware, Unix runs on almost any kind, from tiny embedded systems to giant supercomputers. After you read this book, you'll be ready to use many kinds of computers without learning a new OS for each one.
- In general, Unix (especially without a windowing system) is less resource-intensive than other major operating systems. For instance, Linux will run happily on an old system with a x386 microprocessor and let multiple users share the same computer. (Don't bother trying to use the latest versions of Microsoft Windows on a system that's more than a few years old!) If you need a windowing system, Unix lets you choose from modern feature-rich interfaces as well as from simple ones that need much less system power. Anyone with limited resources—educational institutions, organizations in developing countries, and so on—can use Unix to do more with less.
- Much of the Internet's development was done on Unix systems. Many Internet web sites and Internet service providers use Unix because it's so flexible and inexpensive. With powerful hardware, Unix really shines.

Versions of Unix

There are several versions of Unix. Until a few years ago, there were two main versions: the line of Unix releases that started at AT&T (the latest is System V Release 4), and another from the University of California at Berkeley (the last version was 4.4BSD). Some past and present commercial versions include SunOS, Solaris, SCO Unix, AIX, HP/UX, and ULTRIX. Freely available versions include Linux, NetBSD, and FreeBSD (FreeBSD is based on 4.4BSD-Lite).

Many Unix versions, including System V Release 4, merge earlier AT&T releases with BSD features. The POSIX standard for Unix-like operating systems defines a single interface to Unix. Although advanced features differ among systems, you should be able to use this introductory handbook on any system.

When we write "Unix" in this book, we mean "Unix and its versions" unless we specifically mention a particular version.

Interfaces to Unix

Unix can be used as it originally was, on typewriter-like terminals, from a shell prompt on a command line. (See the section "Examples," later in this chapter.) Most versions of Unix also work with window systems (sometimes called Graphical User Interfaces, or GUIs). These allow each user to have a single screen with multiple windows—including "terminal" windows that act like the original Unix interface. (Chapter 2 explains window system basics.)

Although a window system lets you use Unix without typing text at a shell prompt, we'll spend most of our time on that traditional command-line interface to Unix. Why?

- Every Unix system has a command-line interface. If you know how to use the command line, you'll always be able to use the system.
- If you become a more-advanced Unix user, you'll find that the command line is actually much more flexible than a windowing interface. Unix programs are designed to use together from the command line—as "building blocks"—in an almost infinite number of combinations, to do an infinite number of tasks. No windowing system that we've seen (yet!) has this tremendous power.
- You can launch and close windowing programs from the command line, but windowing programs generally can't affect a command line or programs you run from one.
- Once you learn to use the command line, you can use those same techniques to write *scripts*. These little (or big!) programs automate jobs you'd have to do manually and repetitively with a window system (unless you understand how to program a window system, which is usually a much harder job). See the section "Programming" in Chapter 8 for a brief introduction to scripting.
- In general, text-based interfaces are much easier than GUIs for sight- and hearing-impaired users.

We aren't saying that the command-line interface is right for every situation. For instance, using the Web—with its graphics and links—is usually easier with a GUI web browser. But the command line is the fundamental way to use Unix. Understanding it will let you work on any Unix system, with or without windows.

What This Handbook Covers

This book teaches basic system utility commands to get you started with Unix. Instead of overwhelming you with lots of details, we want you to be comfortable in the Unix environment as soon as possible. So we cover a command's most useful features instead of describing all its options in detail.

We also assume that your computer works properly; someone has started it, knows the procedure for turning the power off, and knows how to perform system maintenance. In other words, we don't cover Unix system administration.

Unix users can choose between many different user interfaces—shells and window systems. Our examples show the **bash** shell and the GNOME and KDE window environments. We've chosen them because they're popular and make good examples, not because we think they're always "the best." If you do advanced work or set up Unix systems for other users, we recommend learning about a variety of shells and window systems and choosing the best ones for your needs. The principles explained in this book should help you use any Unix configuration.

What's New in the Fifth Edition

Unix keeps evolving, and this book changes with it. Although most tips in this book work on all Unix systems, old and new, there have been changes since 1997 that justify a fifth edition. Over the years, readers have asked us to include topics that couldn't be covered in just a few paragraphs—a text editor, for instance. We've decided to let this little book grow just a bit by adding several-page overviews of popular Unix tools: the Pico text editor, the Pine email program, the Lynx web browser, and two interactive chat programs. Networking is much more common, so we've added a new chapter about it. Our windowing examples show newer window systems and you'll find sections about command-line editing. There's a new Glossary with definitions of common terms, and the Index has also been expanded. Finally, we've made changes suggested by our readers.

Format

The following sections describe conventions used in this handbook.

Commands

We introduce each main concept first, and then break it into task-oriented sections. Each section shows the best command to use for a task, explains what it does, and shows the syntax (how to put the command line together). The syntax is given like this:

rm *filename*

Commands appear in **boldface** type (in this example, **rm**). You should type the command exactly as it appears in the example. The variable parts (here, *filename*) will appear in *italic* type; you must supply your own value. To enter this command, you would type **rm** followed by a space and the name of the file that you want to remove, then press the RETURN key. (Your keyboard may have a key labeled ENTER or an arrow with a right-angle shaft instead of a RETURN key.) Throughout this book, the term *enter* means to type a command and press RETURN to run it.

Examples

Examples show what should happen as you enter a command. Some examples assume that you've created certain files. If you haven't, you may not get the results shown.

We use typewriter-style characters for examples. Items you type to try the example are **`boldface`**. System messages and responses are `normal text`.

Here's an example:

```
$ date
Tue Oct  9 13:39:24 MST 2001
$
```

The character "$" is the shell (system) prompt. To do this example, you would type **date** and then press RETURN. The **date** command responds "Tue Oct 9 13:39:24 MST 2001" and then returns you to the prompt.

Text you see in examples may not be exactly what you see on your screen. Different Unix versions have commands with different outputs. Sometimes we edit screen samples to eliminate distracting text or make them fit the page.

Problem Checklist

We've included a problem checklist in some sections. You may skip these parts and go back to them if you have a problem.

Exercises

Some sections have exercises to reinforce text you've read. Follow the exercises, but don't be afraid to experiment on your own.

Exercises have two columns. The lefthand column tells you what to do and the righthand column tells you how to do it. For example, a line in the section "Exercise: entering a few commands," near the end of Chapter 1, shows the following:

Get today's date Enter **`date`**

To follow the exercise, type in the word **date** on your keyboard and then press the RETURN key. The lefthand column tells you what will happen.

After you try the commands, you'll have a better idea of the ones you want to learn more about. You can then get more information from a source in the section "Documentation," in Chapter 8.

Comments and Questions

Please address comments and questions concerning this book to the publisher:

O'Reilly & Associates, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
(800) 998-9938 (in the United States or Canada)
(707) 829-0515 (international or local)
(707) 829-0104 (fax)

To ask technical questions or comment on the book, send email to:

bookquestions@oreilly.com

We have a web site for the book where examples, errata, and any plans for future editions are listed. You can access this site at:

http://www.oreilly.com/catalog/lunix5/

For more information about books, conferences, Resource Centers, and the O'Reilly Network, see the O'Reilly web site at:

http://www.oreilly.com

If you write to us, please include information about your Unix environment and the computer you use. You'll have our thanks, along with thanks from future readers of this handbook.

Acknowledgments

H. Milton Peek reviewed the first draft of this edition. Jeff Kawski acted as the technical editor. Chris Stone of O'Reilly & Associates, Inc. gave information about Mac OS X and reviewed the section about it. And Tim, thanks from Jerry for all your advice and support during my 12 years of writing for O'Reilly.