LEARNING MADE EASY

7th Edition

Linux®

ALL-IN-ONE

for dummies®
A Wiley Brand

8 Books in one!

Richard Blum
Has used Linux since the days when it took 55 floppy disks to load it

Linux®

ALL-IN-ONE

7th Edition

by Richard Blum

for dummies®
A Wiley Brand

Linux® All-in-One For Dummies®, 7th Edition

Published by: **John Wiley & Sons, Inc.,** 111 River Street, Hoboken, NJ 07030-5774, www.wiley.com

For general information on our other products and services, please contact our Customer Care Department within the U.S. at 877-762-2974, outside the U.S. at 317-572-3993, or fax 317-572-4002. For technical support, please visit https://hub.wiley.com/community/support/dummies.

Wiley publishes in a variety of print and electronic formats and by print-on-demand. Some material included with standard print versions of this book may not be included in e-books or in print-on-demand. If this book refers to media such as a CD or DVD that is not included in the version you purchased, you may download this material at http://booksupport.wiley.com. For more information about Wiley products, visit www.wiley.com.

Library of Congress Control Number: 2022945629

ISBN 978-1-119-90192-1 (pbk); ISBN 978-1-119-90194-5 (ebk); ISBN 978-1-119-90193-8 (ebk)

Contents at a Glance

Table of Contents

Introduction

Linux is truly amazing when you consider how it originated and how it continues to evolve. From its modest beginning as the hobby of one person — Linus Torvalds of Finland — Linux has grown into a full-fledged operating system with features that rival those of any commercial Unix operating system. To top it off, Linux — with all its source code — is available free to anyone. All you have to do is download it from a website or get it on a USB flash drive, CD, or DVD for a nominal fee from one of many Linux CD vendors.

Linux certainly is an exception to the rule that "you get what you pay for." Even though Linux is free, it's no slouch when it comes to performance, features, and reliability. The robustness of Linux has to do with the way it is developed and updated. Developers around the world collaborate to add features. Incremental versions are continually downloaded by users and tested in a variety of system configurations. Linux revisions go through much more rigorous beta testing than any commercial software does.

If you're beginning to use Linux, what you need is a practical guide that not only gets you going with Linux installation and setup but also shows you how to use Linux for a specific task. You may also want to try out different Linux distributions before settling on one.

About This Book

Linux All-in-One For Dummies gives you eight quick-reference guides in a single book. Taken together, these eight minibooks provide detailed information on installing, configuring, and using Linux, as well as pointers for passing the vendor-neutral certification exams available from CompTIA and the Linux Professional Institute (LPI) to authenticate your skills.

What you'll like most about this book is that you don't have to sequentially read the whole thing chapter by chapter — or even read through each section in a chapter. You can pretty much turn to the topic you want and quickly get the answer to your pressing questions about Linux, whether they're about using the LibreOffice.org word processor, setting up the Apache web server, or a wide range of topics.

Topics that correspond to the certification objectives are important after you've become comfortable enough with the operating system to consider taking the certification exams. As I discuss the material, Tips draw your attention to the key concepts and topics tested in the CompTIA Linux+ or LPI LPIC-1 exams. Note, though, that not all Tips indicate material that's on the exams; I also share other types of information in Tips.

If you are a novice to Linux, ignore the certification objective information as you read. Only after you become comfortable with the operating system and are considering authenticating your skills by taking the CompTIA or LPI exams should you revisit the book and look for this information.

Each minibook zeros in on a specific task area — such as using the Internet or running Internet servers — and then provides hands-on instructions on how to perform a series of related tasks. You can jump right to a section and read about a specific task. You don't have to read anything but the few paragraphs or the list of steps that relate to your question. Use the Table of Contents or the Index to locate the pages relevant to your question.

You can safely ignore text next to the Technical Stuff icons, as well as text in sidebars. However, if you're the kind of person who likes to know some of the hidden details of how Linux works, then, by all means, dig into the Technical Stuff icons and the sidebars.

Within this book, you may note that some web addresses break across two lines of text. If you're reading this book in print and want to visit one of these web pages, simply key in the web address exactly as it's noted in the text, pretending as though the line break doesn't exist. If you're reading this as an e-book, you've got it easy — just click the web address to be taken directly to the web page.

Foolish Assumptions

I assume that you're familiar with a PC — you know how to turn it on and off and you've dabbled with Windows. (Considering that most new PCs come preloaded with Windows, this assumption is safe, right?) And I assume that you know how to use some Windows applications, such as Microsoft Office.

When installing Linux on your PC, you may want to retain your Windows installations. I assume that you don't mind shrinking the Windows partition to make room for Linux. For this procedure, you can invest in a good disk-partitioning tool or use one of the partitioning tools included with most Linux distributions.

I also assume that you're willing to accept the risk that when you try to install Linux, some things may not quite work. Problems can happen if you have some uncommon types of hardware. If you're afraid of ruining your system, try finding a slightly older, spare PC that you can sacrifice and then install Linux on that PC. Alternatively, you can install a virtual server software package such as Oracle's VirtualBox and install Linux as a virtual machine inside your Windows desktop.

Linux All-in-One Desk Reference For Dummies has eight minibooks, each of which focuses on a small set of related topics. If you're looking for information on a specific topic, check the minibook names on the thumb tabs or consult the Table of Contents.

Icons Used in This Book

Following the time-honored tradition of the *All-in-One For Dummies* series, I use icons to help you quickly pinpoint useful information. The icons include the following:

REMEMBER

The Remember icon marks a general, interesting fact — something that you want to know and remember as you work with Linux. You might even find interesting trivia worth bringing up at an evening dinner party.

TIP

When you see the Tip icon, you're about to read about something you can do to make your job easier. Long after you've finished with the first reading of this book, you can skim the book, looking for only the tips.

WARNING

I use the Warning icon to highlight potential pitfalls. With this icon, I'm telling you: "Watch out! Whatever is being discussed could hurt your system." They say that those who are forewarned are forearmed, so I hope these entities will save you some frustration.

TECHNICAL STUFF

The Technical Stuff icon marks technical information that could be of interest to an advanced user (or those aspiring to be advanced users).

Beyond the Book

In addition to the book you have in your hands, you can access some helpful extra content online. Check out the free Cheat Sheet by going to www.dummies.com and entering **Linux All-in-One For Dummies** in the Search box. You'll find common Linux commands and where to go for more help with Linux.

Occasionally, we have updates to our technology books. If this book does have any technical updates, they'll be posted at www.dummies.com.

Where to Go from Here

It's time to get started on your Linux adventure. Turn to any chapter and let the fun begin. Use the Table of Contents and the Index to figure out where you want to go. Before you know it, you'll become an expert at Linux!

I hope you enjoy consulting this book as much as I enjoyed writing it!

1

Getting Started with Linux

Contents at a Glance

IN THIS CHAPTER

» Getting acquainted with Linux

» Discovering Linux distributions

Chapter 1
Introducing Linux

The Linux operating system has become one of the most widely used operating systems, popular among researchers, application developers, and hobbyists alike. These days, the Linux operating system can be found in an amazing range of computer environments, from mobile phones to satellites.

This chapter examines just what the Linux operating system is and why there are so many different Linux distributions available to choose from. With this information, you can select the right Linux distribution for your environment.

What Is Linux?

If you've never worked with Linux before, you may be confused as to why there are so many different versions of it available. You've most likely come across terms such as *distribution, LiveDVD,* and *GNU* when looking at Linux packages, and you may have been confused. This section takes some of the mystery out of the Linux system for you.

Although people usually refer to the Linux operating system as just "Linux," in reality there are quite a few parts that make up a complete Linux system. The four main parts of a Linux system are

- The Linux kernel
- The GNU utilities
- The user interface
- Application software

Each of these four parts has a specific job in the Linux system. Although each of the parts by itself isn't very useful, put together, they create what people refer to as "Linux." Figure 1-1 shows the basic diagram of how these parts fit together to create the overall Linux system.

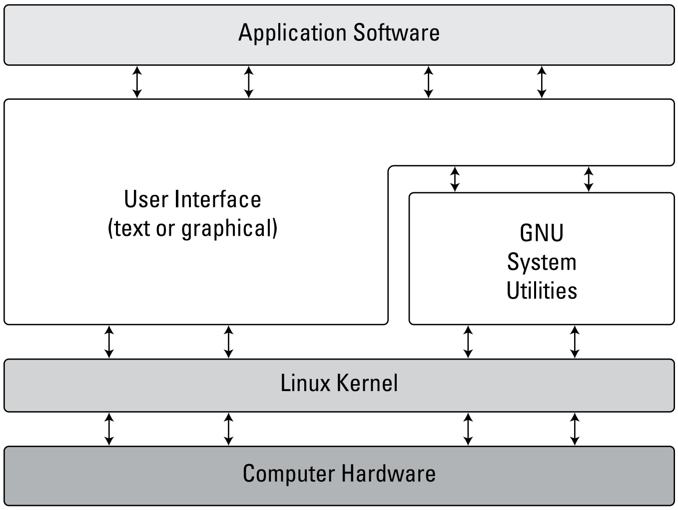

FIGURE 1-1: The Linux system.

The following sections describe these four parts in detail and give you an overview of how they work together to create a complete Linux system.

The Linux kernel

The core of the Linux system is the *kernel.* The kernel controls all the hardware and software on the computer system, allocating hardware when necessary and executing software when required.

If you've been following the Linux world at all, no doubt you've heard the name Linus Torvalds. Linus is the person responsible for creating the first Linux kernel software while he was a student at the University of Helsinki. He intended it to be a copy of the Unix system, at the time a popular operating system used at many universities.

After developing the Linux kernel, Linus released it to the Internet community and solicited suggestions for improving it. This simple process started a revolution in the world of computer operating systems. Soon Linus was receiving suggestions from students as well as professional programmers from around the world.

Allowing anyone to change programming code in the kernel would result in complete chaos. To simplify things, Linus acted as a central point for all improvement suggestions. It was ultimately Linus's decision whether to incorporate suggested code in the kernel. This same concept is still in place with the Linux kernel code, except that instead of just Linus controlling the kernel code, a team of developers has taken on the task.

The kernel is primarily responsible for four main functions:

- System memory management
- Software program management
- Hardware management
- File system management

The following sections explore the first three functions in more detail. File system management in Linux can be somewhat complicated; Book 2, Chapter 5 dives into that topic.

System memory management

One of the primary functions of the operating system kernel is memory management. Memory management is the ability to control how programs and utilities run within the memory restrictions of the system. Not only does the kernel manage the physical memory available on the system, but it can also create and manage *virtual memory* (memory that doesn't actually exist but is created on the hard drive and treated as real memory).

It does this by using space on the hard disk called the *swap space.* The kernel swaps the contents of virtual memory locations back and forth from the swap space to the actual physical memory. This allows the system to think there is more memory available than what physically exists, as shown in Figure 1-2.

The memory locations are grouped into blocks called *pages.* The kernel locates each page of memory either in the physical memory or the swap space. The kernel then maintains a table of the memory pages that indicates which pages are in physical memory and which pages are swapped out to disk.

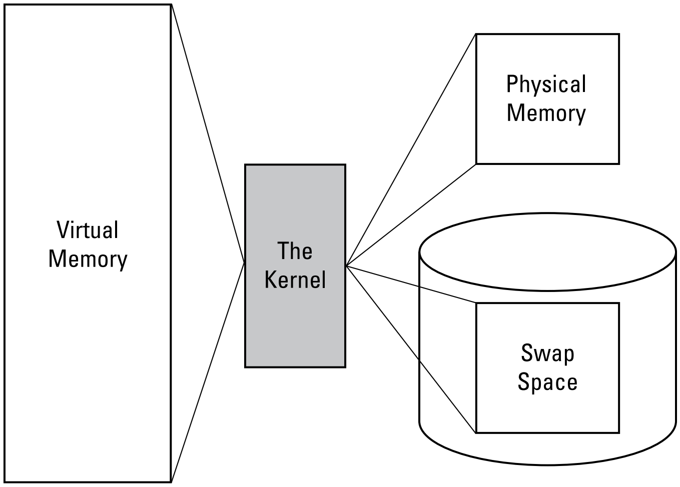

FIGURE 1-2: The Linux system memory map.

The kernel keeps track of which memory pages are in use and automatically copies memory pages that have not been accessed for a period of time to the swap space area (called *swapping out*) — even if other memory is available. When a program wants to access a memory page that has been swapped out, the kernel must make room for it in physical memory by swapping out a different memory page and swapping in the required page from the swap space. Obviously, this process takes time and can slow down a running process. The process of swapping out memory pages for running applications continues for as long as the Linux system is running.

Software program management

With the Linux operating system, a running program is called a *process.* A process can run in the foreground, displaying output on a display, or it can run in the background, doing work behind the scenes. The kernel controls how the Linux system manages all the processes running on the system.

The kernel creates the first process, called the *init process*, to start all other processes on the system. When the kernel starts, it loads the init process into virtual memory. As the kernel starts each additional process, it gives it a unique area in virtual memory to store the data and code that the process uses.

A few different types of init process implementations are available in Linux, but these days, the two most popular are

- **SysVinit:** The SysVinit initialization method was the original method used by Linux and was based on the Unix System V initialization method. Though it is not used by many Linux distributions these days, you still may find it around in older Linux distributions.

- **systemd:** The systemd initialization method was created in 2010 and has become the most popular initialization and process management system used by Linux distributions.

The SysVinit initialization method primarily utilizes scripts to start and stop applications as needed, while the systemd initialization method uses configuration files. Book 4, Chapter 2 explores how each of these initialization methods works and how you can configure them to customize which applications your Linux system starts automatically.

Hardware management

Still another of the kernel's responsibilities is hardware management. Any device that the Linux system must communicate with needs driver code inserted inside the kernel code. The driver code in the kernel allows the kernel to pass data back and forth to the device, acting as a middleman between applications and the hardware. There are two methods used for inserting device driver code in the Linux kernel:

- Drivers compiled in the kernel
- Driver modules added to the kernel during runtime

Originally, the only way to insert device driver code was to recompile the kernel and restart the system. Each time you added a new device to the system, you had to recompile the kernel code and restart. This process became even more inefficient as Linux kernels supported more hardware and as removable storage devices (such as USB sticks) became more popular. Fortunately, Linux developers devised a better method to insert driver code into the running kernel.

Programmers developed the concept of *kernel modules* to allow the insertion of device driver code into a running kernel without having to recompile the kernel. Also, a kernel module could be removed from the kernel when the system had finished using the device. This greatly simplified and expanded using hardware with Linux.

Book 4, Chapter 2 also dives into driver modules and how to use them in your Linux system.

GNU utilities

Besides having a kernel to control memory, software, and hardware devices, a computer operating system needs utilities to perform standard functions, such as handling files and programs. Although Linus created the Linux system kernel, he had no system utilities to run on it. Fortunately for him, at the same time he was working, a group of people were working together on the Internet trying to develop a standard set of computer system utilities that mimicked the popular Unix operating system.

The GNU Project (GNU stands for GNU's Not Unix — a recursive acronym) developed a complete set of Unix utilities but had no kernel system to run them on. These utilities were developed under a software philosophy called open-source software (OSS).

The concept of OSS allows programmers to develop software and then release it to the world with no licensing fees attached. Anyone can use the software, modify it, or incorporate it into their own system without having to pay a license fee. Uniting Linus's Linux kernel with the GNU operating system utilities created a complete, functional, free operating system.

Although the bundling of the Linux kernel and GNU utilities is often just called Linux, you'll see some Linux purists on the Internet refer to it as the GNU/Linux system to give credit to the GNU Project for its contributions to the cause.

The GNU Project was mainly designed for Unix system administrators to have a Unixlike environment available. This focus resulted in the project porting many common Unix system commandline utilities. The core bundle of utilities supplied for Linux systems is called the `coreutils` package.

The GNU `coreutils` package consists of three parts:

- Utilities for handling files
- Utilities for manipulating text
- Utilities for managing processes

Each of these three main groups of utilities contains several utility programs that are invaluable to the Linux system administrator and programmer.

Linux user interfaces

Having a world-class operating system that can manage your computer hardware and software is great, but you need some way to communicate with it. With the popularity of Microsoft Windows, desktop computer users expect some type of graphical display to interact with their system. This spurred more development in the OSS community, and the Linux graphical desktops emerged.

Linux is famous for being able to do things in more than one way, and no place is this more relevant than in graphical desktops. There are a plethora of graphical desktops you can choose from in Linux. The following sections describe a few of the more popular ones.

The X Window system

There are two basic elements that control your video environment: the video card in your workstation and your monitor. To display fancy graphics on your computer, the Linux software needs to know how to talk to both of them.

The X Window software is a lowlevel program that works directly with the video card and monitor in the workstation and controls how Linux applications can present fancy windows and graphics on your computer.

Linux isn't the only operating system that uses X Window; there are versions written for many different operating systems. In the Linux world, there are a few different software packages that can implement it, but there are two that are most commonly used:

- **X.org:** Based on the original Unix X Window System version 11 (often called X11), it's the older of the two packages.
- **Wayland:** More Linux distributions are migrating to the Wayland software, a newer X Window package that is touted to be more secure and easier to maintain.

When you first install a Linux distribution, it attempts to detect your video card and monitor and then creates an X Window configuration file that contains the required information. During installation, you may notice a time when the installation program scans your monitor for supported video modes. Sometimes this causes your monitor to go blank for a few seconds. Because there are lots of different types of video cards and monitors out there, this process can take a little while to complete.

The core X Window software produces a graphical display environment but nothing else. Although this is fine for running individual applications, it isn't too useful for day-to-day computer use. There is no desktop environment allowing users to manipulate files or launch programs. To do that, you need a desktop environment on top of the X Window system software.

The KDE Plasma desktop

The K Desktop Environment (KDE) was first released in 1996 as an open-source project to produce a graphical desktop similar to the Microsoft Windows environment. The KDE desktop incorporates all the features you're probably familiar with if you're a Windows user. Figure 1-3 shows the current version, called KDE Plasma, running in the openSUSE Linux distribution.

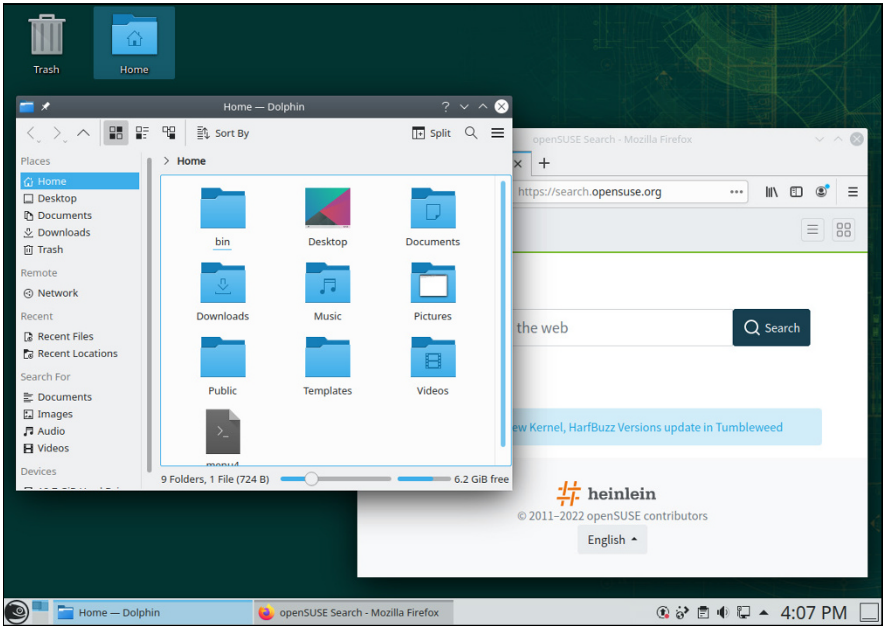

FIGURE 1-3: The KDE Plasma desktop in openSUSE.

The KDE project also maintains lots of common desktop applications that run in KDE Plasma. Book 2, Chapter 2 explores all the features of the KDE Plasma desktop environment.

The GNOME desktop

The GNU Network Object Model Environment (GNOME) is another popular Linux desktop environment. First released in 1999, GNOME has become the default desktop environment for many Linux distributions (the most popular being Red Hat Enterprise Linux).

The GNOME desktop underwent a radical change with version 3, released in 2011. It departed from the standard look and feel of most desktops that use standard menu bars and task bars, and instead made the interface more menu-driven so it would be user-friendly across multiple platforms, such as tablets and mobile phones. This change led to controversy, spawning many new desktops that kept the GNOME 2 look. Figure 1-4 shows the standard GNOME 3 desktop used in the Ubuntu Linux distribution.