chapter 1

Introduction to Linux Toys II

in this chapter

☑ Learning about Linux

☑ Linux Toys II projects

☑ Understanding "free" software

☑ Improvements from Linux Toys to Linux Toys II

The same building blocks used to create the world's most powerful computer system (Linux) can also be used as the foundation for creating fun, interesting, and useful projects. If you have a PC and this book (and are okay with adding a few extra pieces of hardware, which I describe later), you're ready to start.

Welcome to *Linux Toys II.*

Linux Toys II is my second attempt (the first being *Linux Toys*) to gather up a few cool projects based on open source software and create a book that steps through how to make them. In the few years since *Linux Toys* was published, the pool of open source software on which to base *Toys* projects has grown astronomically. As a result, you'll find *Linux Toys II* projects to be both more solid and more engaging than those in the first book. (See the description of the differences between the two books later in this chapter.)

Besides being fun, *Linux Toys II* projects offer you other opportunities:

- **Learn Linux** — While this book might be a bit challenging if you have never used Linux before, I've done what I can to simplify instructions and add some primers (see Appendixes B and C) for using and installing Linux. You might consider getting a more detailed book on installing, configuring, and using Linux (such as *Red Hat Fedora and Enterprise Linux 4 Bible*) to back you up if you are a first-time Linux user.

  For experienced Linux users, I've tried to remove a lot of the hunting and pecking you often have to do to get a project going and to expose you to some areas of Linux you may not have tried before.

- **Exercise your freedom** — By building these computer projects with open source software, you are generally free to add, take apart, recompile, or change out the software elements of these projects without breaking proprietary licensing agreements. (See the description of what you can and can't do with "free" software later in this chapter.)

- The spirit of open source software is sharing the pieces of software you add so that together everyone can benefit. Examples of this spirit include the ease with which you can share virtual worlds in the BZFlag tank game or download selected games, office applications, or network tools to your customized Damn Small Linux pen drive.
- **Reuse an old PC**—While the latest Windows operating system won't run on that eight-year-old PC sitting in your closet, many of the projects in this book will. If you can get an Ethernet card for it, even your old 486 machine can probably be used for the Linux Terminal Server Project.

  Most PCs that can boot from a CD will work for the bootable Linux projects (such as the *Linux Toys II* projects based on eMoviX and the Devil-Linux firewall CD). At the very least, you should be able to boot the *Linux Toys II* CD to try out Damn Small Linux (which includes a graphical interface with streamlined desktop applications).
- **Play with computer hardware**—While some of the projects are purely software-oriented (so they can run on most PCs), a few give you the chance to put together and play with the hardware as well. Follow the instructions in the MythTV project to build a personal video recorder with a wireless keyboard, a remote control, and a slick case that will fit in any entertainment center. Create a firewall box for Devil-Linux that includes the switch for attaching the computers in your home or small office LAN. Add X10 hardware modules around your house wherever you want to interact with lights and appliances using BottleRocket or Heyu software.

To make sure the projects are accurate and backed up by experts, most chapters have been reviewed by the creator or project lead for the featured software. Many of the projects closely follow open source initiatives that have active, on-going development efforts and vibrant community forums and mailing lists. Those facts will help you if you get stuck or want to continue to grow with the *Linux Toys II* projects you build.

Learning About Linux

The projects in *Linux Toys II* reflect the growing phenomenon of open source software and the Linux operating system. For the computer enthusiast or professional, Linux is now in so many places (PCs, enterprise computers, handheld and wireless devices, and so on) that you can hardly ignore it. And if you're going to need it anyway, why not learn about it while you build some toys?

Note

If this is your first exposure to Linux, see the "How Can This Software Be Free?" section later in this chapter. It describes how the licensing for open source software works.

Using Specialized Linux Systems

Freedom to use the software you choose in the way that you want to use it has always been at the heart of Linux and the open source software movement. Because you can take apart all the component parts of Linux and put them back together again as you like, you can end up with Linux systems that include everything you might ever want or only the exact components you need.

Here are some ways in which *Linux Toys II* projects let you use specialized versions of Linux:

- **Multimedia player** — The eMoviX and MoviX^2 projects include a specialized Linux system, geared toward playing multimedia content. Not only does the resulting Linux system not have to be installed on a hard disk, but it is small enough to run from RAM. So you can remove the CD containing Linux and insert a CD containing video, music, or digital images to play.
- **Efficient portable desktop system** — Damn Small Linux (DSL) fits on a bootable business card (about 50MB), but contains enough components to do basic Web browsing, email, word processing, and other desktop activities. (The project based on DSL describes how to install it to a rewriteable pen drive.)
- **Thin client** — While the other special Linux systems just described typically boot from CD or other removable medium, the Linux Terminal Server Project (LTSP) helps you get the Linux system that runs on a client computer from a server on the network. LTSP can be configured to boot from a DHCP server via a network card (using PXE or Etherboot) on a cheap PC or stripped-down workstation.

Each of the projects just described can help you learn about the most basic issues related to booting a Linux system. Once you get familiar with those concepts, you will probably want to begin customizing your own Linux systems to do just what you want.

Working with Servers

Several of the *Linux Toys II* projects describe how to set up and offer particular kinds of services on a network. Sharing services from your computer to the Internet is not something that should be done lightly (I'll give you about a billion warnings before the book is done). However, learning how to set up services from the safety of your home or small office LAN (provided you're behind a sturdy firewall, which I also describe) is a great thing.

Here are some of the server projects you can try out in *Linux Toys II*:

- **Sharing photos** — Web servers can do more than just serve HTML pages. The Gallery project includes a full-featured Web-based interface for controlling and sharing albums of digital images. In this project, you touch on features for setting up a Web server, managing access, and organizing digital content.
- **Multi-user gaming** — While many commercial PC games don't run in Linux, many servers that allow you to play those games against multiple opponents over a network do run in Linux. To play with the concept of hosting your own gaming server, the BZFlag project is a great choice. It includes server and client software for battling tanks in a virtual world.

- **Internet radio station** — You can serve up streaming audio content to make your own Internet radio station using Icecast software. The *Linux Toys II* project covering Icecast describes tools for both creating the content to feed to your Icecast server and the server itself for broadcasting the stream over the Internet.

Video and Audio Recording and Playback

The open source community now has its own ways to compress and store video and audio content. So once you have created your home movies, recorded your favorite TV shows, and saved your music, you can compress and store them using Ogg Vorbis (audio) and Theora (video) open source codecs and tools.

Several of the *Linux Toys II* projects focus on working with audio and video content:

- **Personal video recorder** — With MythTV, you can download a Web-based listing of television shows in your area. From that listing you can select to record those shows either immediately or queue them to record when they come on later. With MythTV you can manage and play back your recorded audio/video files as well.
- **Bootable movies** — With eMoviX, you combine a video/audio file with the movie player and Linux software to create a movie that can boot up and play on most PCs. You can record any type of content supported by the MPlayer multimedia player.
- **Streaming audio** — The Xiph.Org Foundation is a champion of open source multimedia-related projects. Using software sponsored by that project, in particular Ogg Vorbis (for compressing audio) content and Icecast (for streaming it to the Internet or other network), I describe a project for creating your own Internet radio station.

Hardware Tinkering

While most of the projects will run on most PCs, I selected a few projects to incorporate some hardware gathering and tinkering as well. If you want the box to look as sweet as the software, I suggest you check out these projects:

- **Entertainment center PC** — For the MythTV project, Tom Weeks combined a PC case that's slick enough to place in your entertainment center for recording, storing, and playing video content. The hardware Tom put together includes a SilverStone Technology LaScala case, two Hitachi hard drives, a Hauppauge TV capture card (with remote control), an NVidia video card, and a wireless keyboard.
- **Firewall/network switch PC** — A firewall for a small office/home office (SOHO) LAN won't require much of a PC. (Tom uses a P133 with 128MB of RAM that someone might be giving away these days.) However, you can add a few extra hardware components to outfit it like commercial dedicated firewall devices you might purchase. Besides a couple of inexpensive network cards (one facing your LAN and the other facing the Internet), Tom incorporates a cheap 100Mbit switch with at least eight ports. It makes a nice little box when you are done.

Aside from those two projects, most of the other *Linux Toys II* projects can be done with smaller bits of extra hardware. For the X10 projects, you need hardware to connect your PC to your location's power source to send X10 signals (usually from the PC's serial port). Then you can add as many X10 hardware modules around your place as you like. For the bootable pen drive project, you need to pick a pen drive (1GB pen drives are now available for just over $50).

The topics of interest covered in *Linux Toys II* projects that I've just described here are just the tip of the iceberg of what you can do with Linux and open source software. But the skills you learn in building these projects will scale down to handheld devices or up to enterprise computing clusters.

About the Linux Toys II Projects

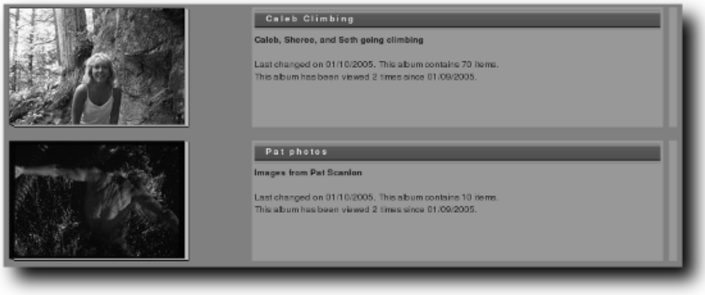
Behind each *Linux Toys II* project are one or two primary open source initiatives. In most cases, these initiatives started because someone had an idea to create something with software. They pulled in existing open source software and created what they needed to fill in the holes.

Here are the *Linux Toys II* projects described in this book and the primary open source efforts the projects are built on.

Web Photo Gallery (Gallery)

A big reason Bharat Mediratta created Gallery was so his mom could easily display and print photos of her grandchildren on the Internet. Today, Gallery is probably the most popular open source software for sharing digital images. The Web Photo Gallery you build in Chapter 3 with Gallery software can be used to organize, manage, display, and print your images from a Web browser over private or public networks (such as the Internet).

Figure 1-1 shows a snippet containing two albums from a Gallery front page.



Caleb Climbing

Caleb, Sheree, and Seth going climbing

Last changed on 01/10/2005. This album contains 70 items.
This album has been viewed 2 times since 01/09/2005.

Pat photos

Images from Pat Scanlon

Last changed on 01/10/2005. This album contains 10 items.
This album has been viewed 2 times since 01/09/2005.

FIGURE 1-1: Share albums of digital images with Gallery.

Personal Video Recorder (MythTV)

Video recording and playback have long been a challenge in Linux and other open source systems. The MythTV project described in Chapter 4 has separated itself from the short list of open source Personal Video Recorders (PVRs) by combining an excellent graphical interface for searching out, selecting, and recording TV shows as well as by including good support for a range of useful recording hardware devices.

Of all the projects described in this book, MythTV is probably the most challenging from both the hardware and software perspective. But if you step through the *Linux Toys* procedure for configuring MythTV, you'll have a fully configurable PVR for recording and storing your television shows or other video input. And you'll end up with a neat new component to go into your entertainment center as well.

Figure 1-2 shows an example of the MythTV PC you build in Chapter 4.

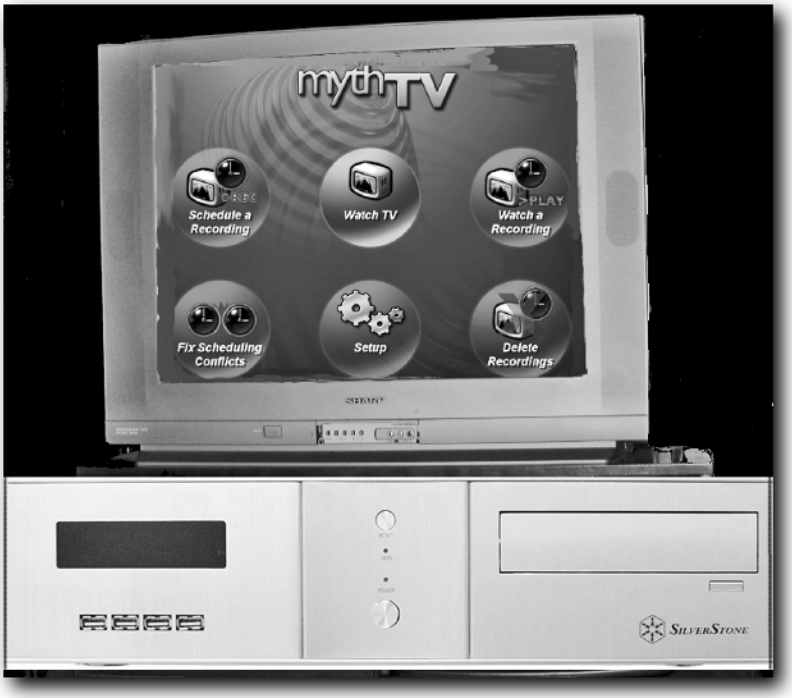


Figure 1-2: Tricking out a SilverStone Lascala case for a MythTV PVR

Bootable Movie Player (eMoviX)

By adding a home movie, a movie player (MPlayer), and a Linux system needed to play the movie to one disk, you can be sure that anyone with a standard PC will be able to play your movies. The *Linux Toys II* project described in Chapter 5 steps you through creating video content in Linux and mastering that content (using eMoviX) to a bootable CD or DVD. The chapter also describes how to use the related MoviX^2 project, which can be run as a bootable player for many different types of multimedia content.

Custom Bootable Pen Drive (Damn Small Linux)

If you like projects that are compact, cool, and contain everything you need, you'll probably find the *Linux Toys II* project in Chapter 6 to be both fun and useful. Using Damn Small Linux (a bootable business card–size Linux that takes up less than 50MB space), the procedure steps you through how to install and configure a fully customized Linux system that you can boot from a pen drive.

Figure 1-3 shows a custom version of Damn Small Linux being booted from a 1GB pen drive, with a personal set of applications, desktop themes, and personal data files (music, documents, digital images, and so on) included on it.

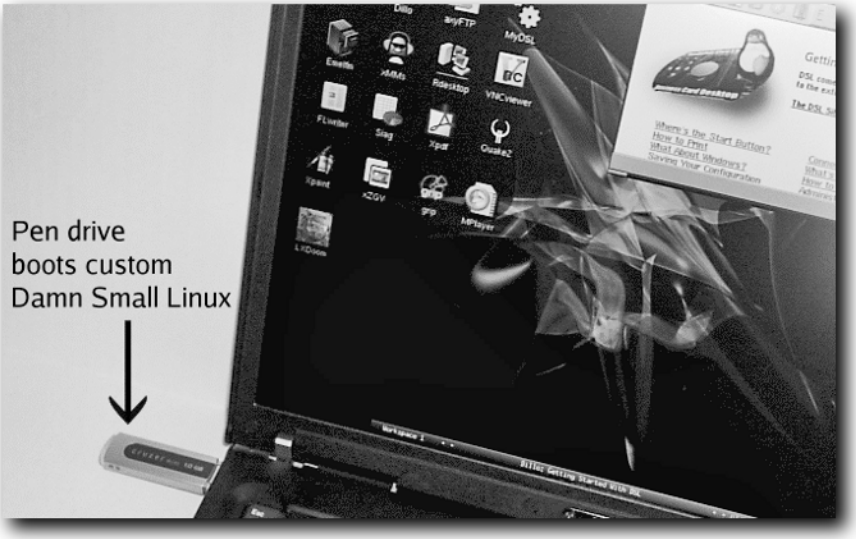


FIGURE 1-3: Boot to your customized Linux desktop from a pen drive.

Personal Firewall (Devil-Linux)

Protecting your workstation or LAN from intruders has been the goal of many different Linux firewall distributions. Devil-Linux, described in Chapter 7, is a 200MB firewall distribution that provides a simple way to configure a secure firewall using a standard PC or by making a firewall PC with a built-in network switch. Figure 1-4 shows a PC that was modified to include a network switch so it could be used as a dedicated firewall.

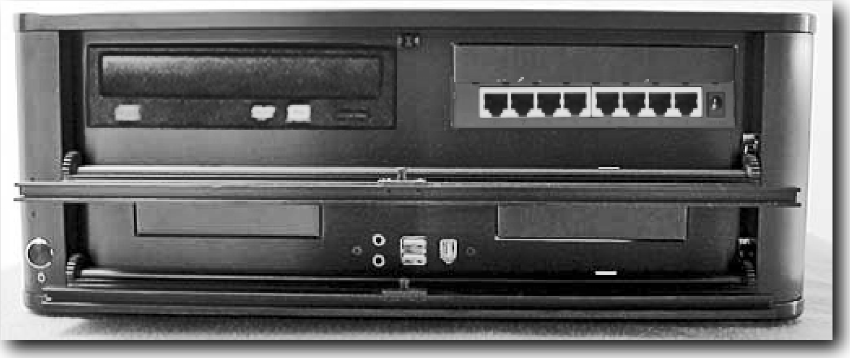


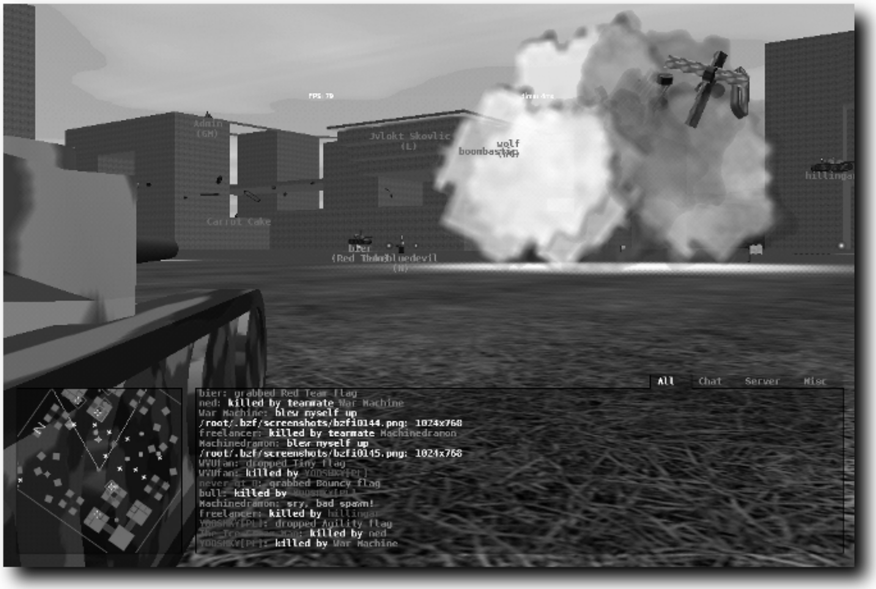
FIGURE 1-4: Create a firewall PC with built-in network switch.

Multi-User Tank Game Server (BZFlag)

Linux is an extremely popular platform for professional and amateur gamers to set up the servers they need to do multi-user, networked gaming. Chris Schoeneman built the BZFlag multi-user, networked tank shoot-em-up game from a graphical demo program he created as a student at Cornell University. Chapter 8 steps you through how to set up a BZFlag server and play the game against multiple opponents on your LAN or the Internet. As shown in Figure 1-5, BZFlag lets you move your tank around a virtual world, picking up flags and destroying opponents.

X10 Home Controller (HeyU and BottleRocket)

Using inexpensive X10 devices, you can control a variety of lights and appliances in and around your home or office. With X10 hardware kits starting at under $50, projects such as BottleRocket and HeyU enable you to dim lights, react to motion detectors, or turn on a sprinkler system from your computer (in real time or using scripts). These X10 projects are described in Chapter 9.



```
bier: grabbed Red Team flag
ned: killed by teammate War Machine
War Machine: blew myself up
/root/.bzf/screenshots/bzfi0144.png: 1024x768
freelancer: killed by teammate Machinedramon
Machinedramon: blew myself up
/root/.bzf/screenshots/bzfi0145.png: 1024x768
WVUfan: dropped Tiny flag
WVUfan: killed by YOOSHKY[PL]
never gt 0: grabbed Bouncy flag
bull: killed by YOOSHKY[PL]
Machinedramon: sry, bad spawn!
freelancer: killed by hillingar
YOOSHKY[PL]: dropped Agility flag
The [illegible]: killed by ned
YOOSHKY[PL]: killed by War Machine
```

FIGURE 1-5: Set up your own BZFlag multi-user, networked tank game.

Internet Radio Station (Icecast)

The people at Xiph.Org that brought you free audio (Ogg Vorbis) and video (Theora) codecs also bring you tools for streaming multimedia content over a network. The Icecast software featured in Chapter 10 can be used to set up streaming audio content so you can broadcast your own music or news radio station on the Internet or other network. Related software from that project called Ices can be used to feed content to your Icecast server, while streaming audio directories on the Internet can be used to advertise the availability of your radio station to the world.

Thin Client Server (Linux Terminal Server Project)

Using old throwaway PCs or stripped-down, diskless workstations, you can fill an entire classroom, small business, or non-profit organization with usable computers. The Linux Terminal Server Project procedure in Chapter 11 steps you through setting up and managing a group of inexpensive workstations from a single LTSP server. The project describes a variety of ways to boot up your clients (such as a CD or from your network card), as well as ways to manage your client computer from your LTSP server.

How Can This Software Be Free?

In recent years, the term "free software" has generally been replaced by the term "open source software" in hopes of clarifying what the movement is all about. Just as "freedom" in society doesn't mean you can do anything you want (such as restrict other people's freedom), freedom with open source software comes with some responsibility. The responsibilities are meant to encourage continued development of free software, when you use the code in certain ways. (See the "Open Source Software Definition" sidebar.)

A programmer who creates open source software typically attaches one of many available open source licenses to it, defining how it can be used. The intention of most open source licenses is to encourage people to make changes to the software and share those changes with others.

Understanding GPL and Other Licenses

The most popular of the open source licenses (and the one that covers most software in Linux distributions) is the GNU General Public License (GPL) from the Free Software Foundation (`www.gnu.org`). You can see that license at the end of this book or at `www.gnu.org/licenses/gpl.html`. As the GPL states, when software authors commit to making their software GPL, the license allows you to:

- Distribute copies of free software (and even charge for this service if you wish)
- Receive source code or get it easily if you want it
- Change the software or use pieces of it in new free programs
- Know you can do these things

There are lots of other licenses that can be used to cover open source–ish kinds of software. The Free Software Foundation maintains a list of some of these licenses (with their own comments about how well they conform to the GNU view of free software) at its Web site (`www.gnu.org/licenses/license-list.html`).

Building Projects with Open Source

As for using Linux and other open source software for building useful and expandable projects, there are several reasons why I think open source software is the best way to go:

- **No licensing fee for each toy** — You can build 1 or 1,000 of each toy. Other than the time you spend, it won't cost you any more than the nothing you paid in the first place (or just the one copy of Fedora Core, Red Hat Enterprise Linux, or other Linux system you bought).
- **No hiding the code** — You can see and change all the code in your *Linux Toys II*. If you don't like what it does, rewrite it yourself.

Open Source Software Definition

The Open Source Definition, written by Bruce Perens, sets down ten points defining open source software (see `www.opensource.org/docs/definition_plain.html` for the complete definition). Here are the points, with my interpretations of what they mean to you as someone creating or using open source software:

- **Free Redistribution**—The software creator can't keep you from selling or giving away the software as part of your software project and the creator can't make you pay a fee for it.
- **Source Code**—The software creator must give you source code or make it available.
- **Derived Works**—The software creator must let you redistribute the software, with your changes, under the same license.
- **Integrity of the Author's Source Code**—If you modify the software, the software creator can ask you to change the name or version, to protect the original code's integrity.
- **No Discrimination Against Persons or Groups**—The software creator can't say, "XYZ people can't use my software because I don't like them."
- **No Discrimination Against Fields of Endeavor**—The software creator can't say, "This software can't be used to study extraterrestrials or create recipes for moonshine."
- **Distribution of License**—Everyone who uses the software can use it under the same license, without needing to add a license.
- **License Must Not Be Specific to a Product**—The software creator can't restrict use of the code to a particular operating system or other software.
- **The License Must Not Restrict Other Software**—The software creator can't restrict you from distributing other software with the open source software. (The example given is that the license can't say that the software can be distributed only with other open source software.)
- **License Must Be Technology-Neutral**—The software creator can't restrict the software in such as way that it must be used with a specific interface style or technology.

- **Get enhancements going forward**—The open source projects that make up the projects described in this book will continue to go forward and offer enhancements to their projects.
- **You can learn Linux**—Every *Linux Toys* project can benefit from the fact that you are building it on a full-service operating system. Learn Linux features for configuring a network connection, a TV capture card, or a Web interface, and you have multiplied the power of your *Linux Toys* project. At the same time, you'll be learning a powerful, professional-quality operating system.

Remember that open source licenses are built on commitments to freedom and community. There are those who would equate people who create or use open source software with thieves and scoundrels. In my experience from Linux mailing lists and my local Linux User Group (LUG), I have found open source devotees as a group to be:

- **The first ones to help you** — Open source supporters always lend a hand if you have a problem with or a question about Linux.
- **Respectful of copyrighted material** — Most open source supporters believe that CDs, DVDs, books, and software should be obtained legally, with proper compensation given to the works' originators. However, they also believe that people should be able to play their legally obtained music and movies on the players they choose (including their PC-based Linux systems).
- **Interested in making things work** — There is a commitment in this community to getting your Linux box to work with any application or computer on your network. Compare that to a company that has a vested interest in selling you more of their products and fewer of the competitor's products. Sometimes they go out of their way to break software that tries to interoperate with their products.

The bottom line is that there are people around who will try to help you overcome obstacles you run into with Linux. Get on a Linux newsgroup or mailing list. Check out the resources at LinuxToys.net. If you can get on the Internet, there's help out there.

There are also other ways to support your own growth with Linux and open source software. For example, if you want to meet other Linux enthusiasts in your area, search out a local Linux User Group (LUG). Many of these groups offer monthly meetings, mailing lists, and Web sites to support their users. To search a list of LUGs for one that is near you, go to the Linux Online User Groups page (`www.linux.org/groups/index.html`).

Improvements from Linux Toys

If you read the first *Linux Toys*, by me and Chuck Wolber (Wiley, 2003), you will notice some improvements in this sequel. Doing that first book was a real learning experience for me and certainly increased my respect for people who run open source projects.

Here are some of the ways that I set out to improve on the approach we took in *Linux Toys*:

- **More depth to each project** — In *Linux Toys II* there is more depth to each project. While in *Linux Toys* we just got each project up and running, here I've taken more time to give you a deeper understanding of the technology behind each project and helped you get it working better.
- **Technical review by project leaders** — Nearly every project has been reviewed by the creator or project lead for the software featured in each project. And as I've asked each project to help us, I've also asked how I should encourage others to help them improve their projects.

- **Tarballs of each project** — *Linux Toys* offered RPMs and source RPMs for each project. *Linux Toys II* includes tarballs of each project as well, to make it easier for those who are installing the projects on Linux systems other than Fedora or Red Hat Enterprise Linux.
- **Bootable Linux Toys CD** — In *Linux Toys II*, the CD that accompanies the book is an actual bootable Linux CD, running Damn Small Linux. This offers several advantages. For one thing, you can use it as the basis for customizing a bootable pen drive project in Chapter 6. However, the CD can also be used to start trying out a reduced Linux system without having to install to hard disk. (Of course, the CD contains the packages needed to install the projects to hard disk as well.)
- **Reliance on other open source initiatives** — By keeping the *Linux Toys II* projects close to (or exactly the same) as the software that is delivered by open source projects, as I did with Gallery, BZFlag, Icecast, and others, readers can go directly to those projects for updates and help. Likewise, if you want to help contribute to these initiatives, there are better mechanisms in place than for the first *Linux Toys*.

In other words, *Linux Toys II* is more of a celebration of existing open source projects than attempts to piece together my own projects. What this book offers is detailed instructions for getting the most out of these very fine open source projects. If you enjoy the projects contained in this book, I strongly encourage you to pursue the people or organizations that created them and find ways to help those open source initiatives grow.

In each chapter, I tell about the major contributors to the software efforts behind each *Linux Toys II project* and what you can do to support or become more involved to help improve the software into the future. So if, in the course of using *Linux Toys II* you find that you have something to add or questions to ask, you will be able to find out just where to go.

Summary

Making useless, old PCs useful again, learning stuff about Linux, and just having fun playing with computer software and hardware are some of the benefits you'll get from the *Linux Toys II* projects contained in this book. The projects in this book cover a range of elements, including bootable Linux systems, open source audio and video playing and streaming, and Linux server configurations.

Projects in *Linux Toys II* have been reviewed directly by creators and maintainers of the open source software each project covers. Because *Linux Toys II* projects stick closely to these popular, well-supported open source initiatives, you have the advantage of having a place to continue to grow your *Linux Toys II* projects beyond the boundaries of this book.