

8

LINUX GUIs: X AND GNOME

IN THIS CHAPTER

X Window System	268
Starting X from a Character-Based Display	270
Remote Computing and Local Displays	270
Desktop Environments/Managers	275
The Nautilus File Browser Window	276
The Nautilus Spatial View	282
GNOME Utilities	284
Run Application Window	286
GNOME Terminal Emulator/Shell	287

This chapter covers the Linux graphical user interface (GUI). It continues where Chapter 4 left off, going into more detail about the X Window System, the basis for the Linux GUI. It presents a brief history of GNOME and KDE and discusses some of the problems and benefits of having two major Linux desktop environments. The section on the Nautilus File Browser covers the View and Side panes, the control bars, the menubar, and the Spatial view. The final section explores some GNOME utilities, including Terminal, the GNOME terminal emulator.

X WINDOW SYSTEM

History of X The X Window System (www.x.org) was created in 1984 at the Massachusetts Institute of Technology (MIT) by researchers working on a distributed computing project and a campuswide distributed environment, called Project Athena. This system was not the first windowing software to run on a UNIX system, but it was the first to become widely available and accepted. In 1985, MIT released X (version 9) to the public, for use without a license. Three years later, a group of vendors formed the X Consortium to support the continued development of X, under the leadership of MIT. By 1998, the X Consortium had become part of the Open Group. In 2001, the Open Group released X version 11, release 6.6 (X11R6.6).

The X Window System was inspired by the ideas and features found in earlier proprietary window systems but is written to be portable and flexible. X is designed to run on a workstation, typically attached to a LAN. The designers built X with the network in mind. If you can communicate with a remote computer over a network, running an X application on that computer and sending the results to a local display is straightforward.

Although the X protocol has remained stable for a long time, additions to it in the form of extensions are quite common. One of the most interesting—albeit one that has not yet made its way into production—is the Media Application Server, which aims to provide the same level of network transparency for sound and video that X does for simple windowing applications.

XFree86 and X.org Many distributions of Linux used the XFree86 X server, which inherited its license from the original MIT X server, through release 4.3. In early 2004, just before the release of XFree86 4.4, the XFree86 license was changed to one that is more restrictive and not compatible with the GPL (page 6). In the wake of this change, a number of distributions abandoned XFree86 and replaced it with an X.org X server that is based on a pre-release version of XFree86 4.4, which predates the change in the XFree86 license. Ubuntu uses the X.org X server, named X; it is functionally equivalent to the one distributed by XFree86 because most of the code is the same. Thus modules designed to work with one server work with the other.

The X stack The Linux GUI is built in layers (Figure 8-1). The bottom layer is the kernel, which provides the basic interfaces to the hardware. On top of the kernel is the X server, which is responsible for managing windows and drawing basic graphical primitives such as lines and bitmaps. Rather than directly generating X commands, most programs use Xlib, the next layer, which is a standard library for interfacing with an X server. Xlib is complicated and does not provide high-level abstractions, such as buttons and text boxes. Rather than using Xlib directly, most programs rely on a toolkit that provides high-level abstractions. Using a library not only makes programming easier, but also brings consistency to applications.

In recent years, the popularity of X has grown outside the UNIX community and extended beyond the workstation class of computers it was originally conceived for. Today X is available for Macintosh computers as well as for PCs running Windows.

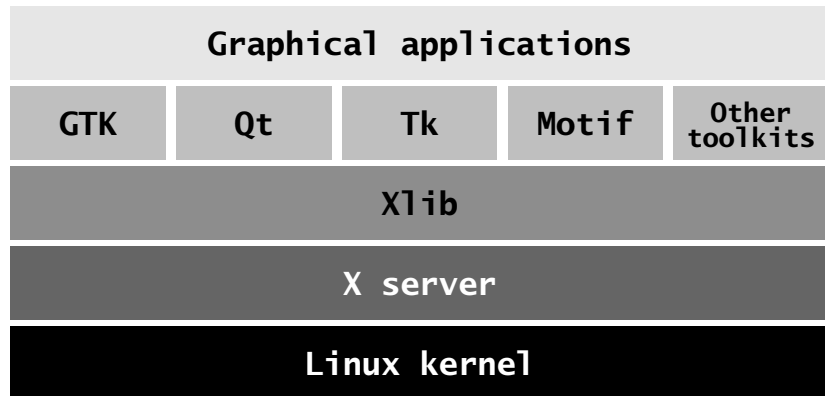


Figure 8-1 The X stack

Client/server environment Computer networks are central to the design of X. It is possible to run an application on one computer and display the results on a screen attached to a different computer; the ease with which this can be done distinguishes X from other window systems available today. Thanks to this capability, a scientist can run and manipulate a program on a powerful supercomputer in another building or another country and view the results on a personal workstation or laptop computer. For more information refer to “Remote Computing and Local Displays” on page 270.

When you start an X Window System session, you set up a *client/server environment*. One process, called the *X server*, displays a desktop and windows under X. Each application program and utility that makes a request of the X server is a *client* of that server. Examples of X clients include `xterm`, `Compiz`, `gnome-calculator`, and such general applications as word processing and spreadsheet programs. A typical request from a client is to display an image or open a window.

The roles of X client and server may be counterintuitive

tip The terms *client* and *server*, when referring to X, have the opposite meanings of how you might think of them intuitively: The server runs the mouse, keyboard, and display; the application program is the client.

This disparity becomes even more apparent when you run an application program on a remote system. You might think of the system running the program as the server and the system providing the display as the client, but in fact it is the other way around. With X, the system providing the display is the server, and the system running the program is the client.

Events The server also monitors keyboard and mouse actions (*events*) and passes them to the appropriate clients. For example, when you click the border of a window, the server sends this event to the window manager (client). Characters you type into a terminal emulation window are sent to that terminal emulator (client). The client takes appropriate action when it receives an event—for example, making a window active or displaying the typed character on the server.

Separating the physical control of the display (the server) from the processes needing access to the display (the client) makes it possible to run the server on one computer and the client on another computer. Most of the time, this book discusses running the X server and client applications on a single system. “Remote Computing and Local Displays” describes using X in a distributed environment.

optional You can run `xev` (X event) by giving the command `xev` from a terminal emulator window and then watch the information flow from the client to the server and back again. This utility opens the Event Tester window, which has a box in it, and asks the X server to send it events each time anything happens, such as moving the mouse pointer, clicking a mouse button, moving the mouse pointer into the box, typing, or resizing the window. The `xev` utility displays information about each event in the window you opened it from. You can use `xev` as an educational tool: Start it and see how much information is processed each time you move the mouse. Close the Event Tester window to exit from `xev`.

USING X

This section provides basic information about starting and configuring X from the command line. For more information see the Xserver man page and the man pages listed at the bottom of the Xserver man page.

STARTING X FROM A CHARACTER-BASED DISPLAY

Once you have logged in on a virtual console (page 149), you can start an X Window System server by using `startx`. See “`rc-sysinit` task and `inittab`” on page 439 for information on creating a `/etc/inittab` file that causes Linux to boot into recovery (single-user) mode, where it displays a textual interface. When you run `startx`, the X server displays an X screen, using the first available virtual console. The following command causes `startx` to run in the background so you can switch back to this virtual console and give other commands:

```
$ startx &
```

REMOTE COMPUTING AND LOCAL DISPLAYS

Typically the X server and the X client run on the same machine. To identify a remote X server (display) an X application (client) is to use, you can either set a global shell variable or use a command-line option. Before you can connect to a remote X server, you must turn off two security features: You must turn off the X `-nolisten tcp` option on the server and you must run `xhost` on the server to give the client permission to connect to the X server. Unless you have a reason to leave these features off, turn them back on when you finish with the examples in this section—leaving them off weakens system security. These tasks must be performed on the X server because the features protect the server. You do not have to prepare the

client. The examples in this section assume the server is named **tiny** and the client is named **dog**.

Security and the Xorg `-nolisten tcp` option

security In a production environment, if you need to place an X server and the clients on different systems, it is best to forward (tunnel) X over ssh. This setup provides a secure, encrypted connection. The method described in this section is useful on local, secure networks and for understanding how X works. See “Forwarding X11” on page 681 for information on setting up ssh so it forwards X.

THE X `-nolisten tcp` OPTION

As Ubuntu is installed, the X server starts with the `-nolisten tcp` option, which protects the X server by preventing TCP connections to the X server. To connect to a remote X server, you must turn this option off on the server. To turn it off, while working with **root** privileges create a file named `/etc/gdm/custom.conf` with the following lines:

```
max@tiny:~$ cat /etc/gdm/custom.conf
[security]
DisallowTCP=false
```

Reboot the system to restart the X server and gdm (gdm-binary) to effect this change. See library.gnome.org/admin/gdm/2.28/configuration.html.en#daemonconfig for more information.

xhost GRANTS ACCESS TO A DISPLAY

As installed, xhost protects each user’s X server. A user who wants to grant access to his X server needs to run xhost. Assume Max is logged in on the system named **tiny** and wants to allow a user on **dog** to use his display (X server). Max runs this command:

```
max@tiny:~$ xhost +dog
dog being added to access control list
max@tiny:~$ xhost
access control enabled, only authorized clients can connect
INET:dog
```

Without any arguments, xhost describes its state. In the preceding example, **INET** indicates an IPv4 connection. If Max wants to allow all systems to access his display, he can give the following command:

```
$ xhost +
access control disabled, clients can connect from any host
```

If you frequently work with other users via a network, you may find it convenient to add an xhost line to your `.bash_profile` file (page 293)—but see the tip on the next page regarding security and xhost. Be selective in granting access to your X display with xhost; if another system has access to your display, you may find your work frequently interrupted.

Security and xhost

security Giving a remote system access to your display using `xhost` means any user on the remote system can watch everything you type in a terminal emulation window, including passwords. For this reason, some software packages, such as the Tcl/Tk development system (www.tcl.tk), restrict their own capabilities when `xhost` permits remote access to the X server. If you are concerned about security or want to take full advantage of systems such as Tcl/Tk, you should use a safer means of granting remote access to your X session. See the `xauth` man page for information about a more secure replacement for `xhost`.

THE DISPLAY VARIABLE

The most common method of identifying a display is to use the `DISPLAY` shell environment variable to hold the X server ID string. This locally unique identification string is automatically set up when the X server starts. The `DISPLAY` variable holds the screen number of a display:

```
$ echo $DISPLAY
:0.0
```

The format of the complete (globally unique) ID string for a display is

```
[hostname]:display-number[.screen-number]
```

where *hostname* is the name of the system running the X server, *display-number* is the number of the logical (physical) display (0 unless multiple monitors or graphical terminals are attached to the system, or if you are running X over ssh), and *screen-number* is the logical number of the (virtual) terminal (0 unless you are running multiple instances of X). When you are working with a single physical screen, you can shorten the identification string. For example, you can use `tiny:0.0` or `tiny:0` to identify the only physical display on the system named `tiny`. When the X server and the X clients are running on the same system, you can shorten this identification string even further to `:0.0` or `:0`. An ssh connection shows `DISPLAY` as `localhost:10.0`. You may have to use `ssh -X` to see this value. See “X11 forwarding” on page 664 for information on setting up ssh so that it forwards X.

If `DISPLAY` is empty or not set, the screen you are working from is not running X. An application (the X client) uses the value of the `DISPLAY` variable to determine which display, keyboard, and mouse (collectively, the X server) to use. One way to run an X application, such as `gnome-calculator`, on the local system but have it use the X display on a remote system is to change the value of the `DISPLAY` variable on the client system so it identifies the remote X server.

```
sam@dog:~$ export DISPLAY=tiny:0.0
sam@dog:~$ gnome-calculator &
```

The preceding example shows Sam running `gnome-calculator` with the default X server running on the system named `tiny`. After setting the `DISPLAY` variable to the ID of the `tiny` server, all X programs (clients) Sam starts use `tiny` as their server (i.e., output appears on `tiny`'s display and input comes from `tiny`'s keyboard and mouse). Try running `xterm` in place of `gnome-calculator` and see which keyboard it accepts input from. If this example generates an error, refer to the two preceding sections, which explain how to set up the server to allow a remote system to connect to it.

When you change the value of DISPLAY

tip When you change the value of the **DISPLAY** variable, all X programs send their output to the new display named by **DISPLAY**.

THE `-display` OPTION

For a single command, you can usually specify the X server on the command line:

```
sam@dog:~$ gnome-calculator -display tiny:0.0
```

Many X programs accept the `-display` option. Those that do not accept this option send their output to the display specified by the **DISPLAY** variable.

RUNNING MULTIPLE X SERVERS

You can run multiple X servers on a single system. The most common reason for running a second X server is to use a second display that allocates a different number of bits to each screen pixel (uses a different *color depth* [page 1141]). The possible values are 8, 16, 24, and 32 bits per pixel. Most X servers available for Linux default to 24 or 32 bits per pixel, permitting the use of millions of colors simultaneously. Starting an X server with 8 bits per pixel permits the use of any combination of 256 colors at the same time. The maximum number of bits per pixel allowed depends on the computer graphics hardware and X server. With fewer bits per pixel, the system has to transfer less data, possibly making it more responsive. In addition, many games work with only 256 colors.

When you start multiple X servers, each must have a different ID string. The following command starts a second X server:

```
$ startx -- :1
```

The `--` option marks the end of the `startx` options and arguments. The `startx` script uses the arguments to the left of this option and passes arguments to the right of this option to the X server. When you give the preceding command in a graphical environment, such as from a terminal emulator, you must work with **root** privileges; you will initiate a privileged X session. The following command starts a second X server running at 16 bits per pixel:

```
$ startx -- :1 -depth 16 &
```

“Using Virtual Consoles” on page 149 describes how to switch to a virtual console to start a second server where you do not have to work with **root** privileges.

Guest Session When you click the Session Indicator (Figure 4-2, page 101), select Guest Session and Ubuntu starts a second X server to accommodate the guest user. When the guest user logs off, the original X server displays the first user’s desktop. You can switch between the X servers (and users) by selecting the virtual console (page 149) that displays the X server you want to work with.

X over ssh See “Tunneling/Port Forwarding” on page 681 for information about running X over an `ssh` connection.

STOPPING THE X SERVER

How you terminate a window manager depends on which window manager is running and how it is configured. If X stops responding, switch to a virtual terminal, log in from

another terminal or a remote system, or use `ssh` to access the system. Then kill (page 455) the process running X. You can also press `CONTROL-ALT-BACKSPACE` to quit the X server. This method may not shut down the X session cleanly; use it only as a last resort.

REMAPPING MOUSE BUTTONS

Throughout this book, each description of a mouse click refers to the button by its position (left, middle, or right, with left implied when no button is specified) because the position of a mouse button is more intuitive than an arbitrary name or number. X numbers buttons starting at the left and continuing with the mouse wheel. The buttons on a three-button mouse are numbered 1 (left), 2 (middle), and 3 (right). A mouse wheel, if present, is numbered 4 (rolling it up) and 5 (rolling it down). Clicking the wheel is equivalent to clicking the middle mouse button. The buttons on a two-button mouse are 1 (left) and 2 (right).

If you are right-handed, you can conveniently press the left mouse button with your index finger; X programs take advantage of this fact by relying on button 1 for the most common operations. If you are left-handed, your index finger rests most conveniently on button 2 or 3 (the right button on a two- or three-button mouse).

“Mouse Preferences” on page 105 describes how to use a GUI to change a mouse between right-handed and left-handed. You can also change how X interprets the mouse buttons using `xmodmap`. If you are left-handed and using a three-button mouse with a wheel, the following command causes X to interpret the right button as button 1 and the left button as button 3:

```
$ xmodmap -e 'pointer = 3 2 1 4 5'
```

Omit the 4 and 5 if the mouse does not have a wheel. The following command works for a two-button mouse without a wheel:

```
$ xmodmap -e 'pointer = 2 1'
```

If `xmodmap` displays a message complaining about the number of buttons, use the `xmodmap -pp` option to display the number of buttons X has defined for the mouse:

```
$ xmodmap -pp
There are 9 pointer buttons defined.
```

Physical Button	Button Code
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9

Then expand the previous command, adding numbers to complete the list. If the `-pp` option shows nine buttons, give the following command:


```
$ xmodmap -e 'pointer = 3 2 1 4 5 6 7 8 9'
```

Changing the order of the first three buttons is critical to making the mouse suitable for a left-handed user. When you remap the mouse buttons, remember to reinterpret the descriptions in this book accordingly. When this book asks you to click the left button or does not specify which button to click, use the right button, and vice versa.

DESKTOP ENVIRONMENTS/MANAGERS

Conceptually X is very simple. As a consequence, it does not provide some of the more common features found in GUIs, such as the ability to drag windows. The UNIX/Linux philosophy is one of modularity: X relies on a window manager, such as Metacity or Compiz, to draw window borders and handle moving and resizing operations.

Unlike a window manager, which has a clearly defined task, a desktop environment (manager) does many things. In general, a desktop environment, such as GNOME or KDE, provides a means of launching applications and utilities, such as a file manager, that work with a window manager.

GNOME AND KDE

The KDE project began in 1996, with the aim of creating a consistent, user-friendly desktop environment for free UNIX-like operating systems. KDE is based on the Qt toolkit made by Trolltech. When KDE development began, the Qt license was not compatible with the GPL (page 6). For this reason the Free Software Foundation decided to support a different project, the GNU Network Object Model Environment (GNOME). More recently Qt has been released under the terms of the GPL, eliminating part of the rationale for GNOME's existence.

GNOME GNOME is the default desktop environment for Ubuntu Linux. It provides a simple, coherent user interface that is suitable for corporate use. GNOME uses GTK for drawing widgets. GTK, developed for the GNU Image Manipulation Program (gimp), is written in C, although bindings for C++ and other languages are available.

GNOME does not take much advantage of its component architecture. Instead, it continues to support the traditional UNIX philosophy of relying on many small programs, each of which is good at doing a specific task.

KDE KDE is written in C++ on top of the Qt framework. KDE tries to use existing technology, if it can be reused, but creates its own if nothing else is available or if a superior solution is needed. For example, KDE implemented an HTML rendering engine long before the Mozilla project was born. Similarly, work on KOffice began a long time before StarOffice became the open-source OpenOffice.org. In contrast, the GNOME office applications are stand-alone programs that originated outside the GNOME project. KDE's portability is demonstrated by the use of most of its core components, including Konqueror and KOffice, under Mac OS X.

Interoperability Since the release of version 2, the GNOME project has focused on simplifying the user interface, removing options where they are deemed unnecessary, and aiming

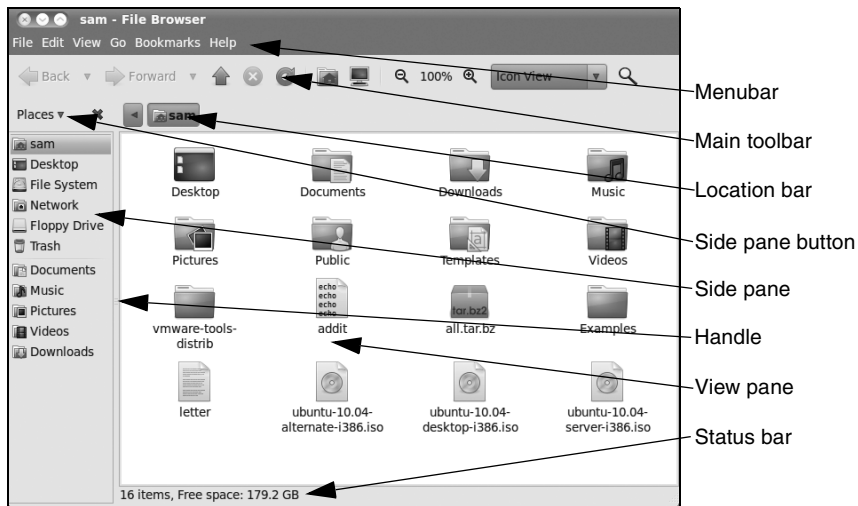


Figure 8-2 A Nautilus File Browser window displaying icons

for a set of default settings that the end user will not wish to change. KDE has moved in the opposite direction, emphasizing configurability.

The freedesktop.org group (freedesktop.org), whose members are drawn from the GNOME and KDE projects, is improving interoperability and aims to produce standards that will allow the two environments to work together. One standard released by freedesktop.org allows applications to use the notification area of either the GNOME or KDE panel without being aware of which desktop environment they are running in.

GNUSTEP

The GNUStep project (www.gnustep.org), which began before both the KDE and GNOME projects, is creating an open-source implementation of the OPENSTEP API and desktop environment. The result is a very clean and fast user interface.

The default look of WindowMaker, the GNUStep window manager, is somewhat dated, but it supports themes so you can customize its appearance. The user interface is widely regarded as one of the most intuitive found on a UNIX platform. Because GNUStep has less overhead than GNOME and KDE, it runs better on older hardware. If you are running Linux on hardware that struggles with GNOME and KDE or if you would prefer a user interface that does not attempt to mimic Windows, try GNUStep. WindowMaker is provided in the `wmaker` package.

THE NAUTILUS FILE BROWSER WINDOW

“Using Nautilus to Work with Files” on page 107 presented an introduction to using Nautilus. This section discusses the Nautilus File Browser window in more depth.

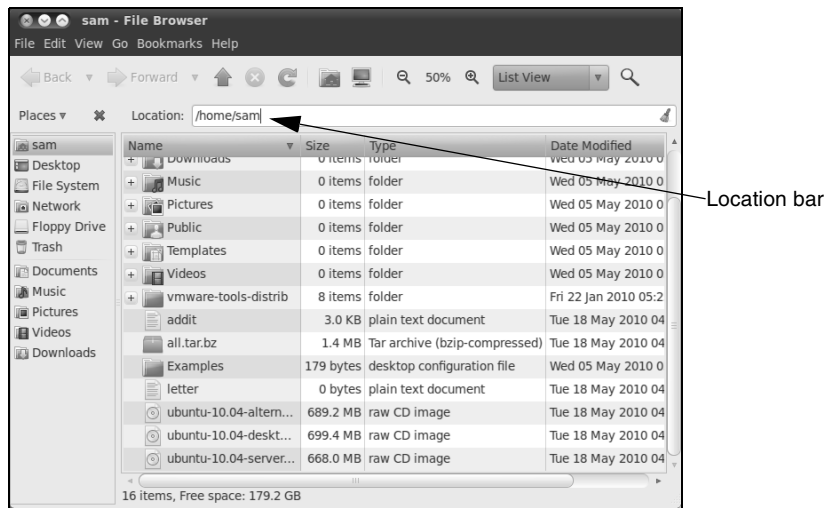


Figure 8-3 A Nautilus File Browser window displaying a List view and a textual location bar

Figure 8-2 shows a File Browser window with a Side pane (sometimes called a *side-bar*), View pane, menubar, toolbar, location bar, and status bar. To display your home folder in a File Browser window, select **Main menu: Places**⇒**Home Folder**.

THE VIEW PANE

The View pane displays icons or a list of filenames. Select the view you prefer from the drop-down list at the right end of the location bar. Figure 8-2 shows an Icon view and Figure 8-3 shows a List view. A Compact view is also available. Objects in the View pane behave exactly as objects on the desktop do. See the sections starting on page 101 for information on working with objects.

You can cut/copy and paste objects within a single View pane, between View panes, or between a View pane and the desktop. The Object context menu (right-click) has cut, copy, and paste selections. Alternatively, you can use the clipboard (page 124) to cut/copy and paste objects.

Nautilus can open a terminal emulator

tip When you install the **nautilus-open-terminal** package (see page 519 for instructions) and log out and log back in, Nautilus presents an Open in Terminal selection in context menus where appropriate. For example, with this package installed, when you right-click a folder (directory) object and select **Open in Terminal**, Nautilus opens a terminal emulator with that directory as the working directory (page 204).

THE SIDE PANE

The Side pane augments the information Nautilus displays in the View pane. Press F9 or click the X at the top of the Side pane to close it. You can display the Side pane by

pressing **F9** or by selecting **File Browser menubar: View⇒Side Pane**. To change the horizontal size of the Side pane, drag the handle (Figure 8-2, page 276) on its right side.

The Side pane can display six types of information. The button at its top controls which type it displays. This button is initially labeled **Places**; click it to display the Side pane drop-down list, which has the selections described next.

Places Places lists folders. Double-click one of these folders to display that folder in the View pane. You can open a directory in a new File Browser window by right-clicking the directory in Places and selecting **Open in New Window**. Right-click and select **Open in New Tab** to open the directory in a new tab.

Places contains two parts: The list above the divider is static and holds your home directory, your desktop, the filesystem, the network, a CD-ROM drive (when it contains a disk), unmounted filesystems (if present), and the trash. The list below the divider holds bookmarks. Add a bookmark by displaying the directory you want to bookmark in the View pane and pressing **CONTROL-D** or by selecting **File Browser menubar: Bookmarks⇒Add Bookmark**. Remove a bookmark by selecting **File Browser menubar: Bookmarks⇒Edit Bookmarks** or by right-clicking the bookmark and selecting **Remove**. You can also use **Edit Bookmarks** to reorder bookmarks.

Information Information presents information about the folder displayed by or highlighted in the View pane.

Tree Tree presents an expandable tree view of your home folder and each mounted filesystem. Each directory in the tree has a plus (+) or minus (–) sign to its left. Click a plus sign to expand a directory; click a minus sign to close a directory. Click a directory in the tree to display that directory in the View pane. Double-click a directory to expand it in the Side pane and display it in the View pane.

History History displays a chronological list of the folders that have been displayed in the View pane, with the most recently displayed folder at the top. Double-click a folder in this list to display it in the View pane.

Notes Notes provides a place to keep notes about the folder displayed in the View pane.

Emblems Similar to the Emblems tab in the Object Properties window (page 129), Emblems allows you to drag emblems from the Side pane and drop them on objects in the View pane. Drag and drop the **Erase** emblem to erase emblems associated with an object. You cannot erase emblems that Ubuntu places on objects, such as locked and link emblems.

CONTROL BARS

This section discusses the four control bars that initially appear in a File Browser window: the status bar, menubar, Main toolbar, and location bar (Figure 8-2, page 276). From **File Browser menubar: View**, you can choose which of these bars to display—except for the menubar, which Nautilus always displays.

- Menubar** The menubar appears at the top of the File Browser window and displays a menu when you click one of its selections. Which menu selections Nautilus displays depend on what the View pane is displaying and which objects are selected. The next section describes the menubar in detail.
- Main toolbar** The Main toolbar appears below the menubar and holds navigation tool icons: Back, Forward, Up, Stop, Reload, Home, Computer, Magnification, View, and Search. If the Main toolbar is too short to hold all icons, Nautilus displays a button with a triangle pointing down at the right end of the toolbar. Click this button to display a drop-down list of the remaining icons.
- To change the magnification of the display in the View pane, click the plus or minus sign in a magnifying glass on either side of the magnification percentage. Right-click the magnification percentage itself to return to the default magnification. Left-click the magnification percentage to display a drop-down list of magnifications. Click the button to the right of the right-hand magnifying glass to choose whether to view files as icons, as a list, or in compact format. Click the magnifying glass at the right end of the toolbar to change the Location bar into a search text box.
- Location bar** Below the Main toolbar is the location bar, which displays the name of the directory that appears in the View pane. It can display this name in two formats: iconic (using buttons) and textual (using a text box). Press **CONTROL-L** to switch to textual format. When you display a different directory in the View pane, Nautilus changes the Location bar back to iconic format.
- In iconic format, each button represents a directory in a pathname (page 205). The View pane displays the directory of the depressed (darker) button. Click one of these buttons to display that directory. If the leftmost button holds a triangle that points to the left, Nautilus is not displaying buttons for all the directories in the absolute (full) pathname; click the button with a triangle in it to display more directory buttons.
- In textual format, the text box displays the absolute pathname of the displayed directory. To have Nautilus display another directory, enter the pathname of the directory and press **RETURN**.
- Status bar** If no items are selected, the status bar, at the bottom of the window, indicates how many items are displayed in the View pane. If the directory you are viewing is on the local system, it also tells you how much free space is available on the device that holds the directory displayed by the View pane. If an item is selected, the status bar displays the name of the item and its size.

MENUBAR

The Nautilus File Browser menubar controls which information the File Browser displays and how it displays that information. Many of the menu selections duplicate controls found elsewhere in the File Browser window. This section highlights some of



Figure 8-4 The Connect to Server window

the selections on the menubar; click **Help** on the menubar and select **Contents** or **Get Help Online** for more information. The menubar holds the menus described next.

- File** The several **Open** selections and the **Property** selection of **File** work with the highlighted object(s) in the **View** pane. If no objects are highlighted, these selections are grayed out or absent. Selecting **Connect to Server** (also available from **Main menu: Places**) displays the **Connect to Server** window (Figure 8-4). This window presents a **Service type** drop-down list that allows you to select **FTP**, **SSH**, **Windows**, or other types of servers. Enter the URL of the server in the text box labeled **Server**. For an **FTP** connection, do not enter the **ftp://** part of the URL. Fill in the optional information as appropriate. Click **Connect**. If the server requires authentication, **Nautilus** displays a window in which you can enter a username and password. **Nautilus** opens a window displaying a directory on the server and an object, named for the URL you specified, on the desktop. After you close the window, you can open the object to connect to and display a directory on the server.
- Edit** Many of the **Edit** selections work with highlighted object(s) in the **View** pane; if no objects are highlighted, these selections are grayed out or absent. This section discusses three selections from **Edit**: **Compress**, **Backgrounds and Emblems**, and **Preferences**.

The **Edit**⇨**Compress** selection creates a single archive file comprising the selected objects. This selection opens a **Compress** window (Figure 8-5) that allows you to specify the name and location of the archive. The drop-down list to the right of the text box labeled **Filename** allows you to specify a filename extension that determines the type of archive this tool creates. For example, **.tar.gz** creates a **tar** (page 176) file compressed by **gzip** (page 175) and **.tar.bz2** creates a **tar** file compressed by **bzip2** (page 174). Click the plus sign to the left of **Other Objects** to specify a password for and/or to encrypt the archive (available only with certain types of archives). You can also split the archive into several files (volumes).

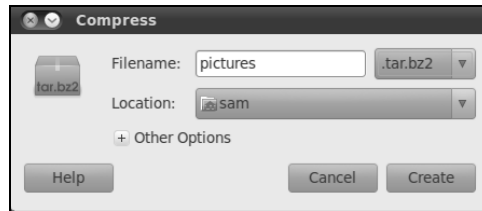


Figure 8-5 The Compress window

The **Edit⇒Backgrounds and Emblems** selection has three buttons on the left: Patterns, Colors, and Emblems. Click **Patterns** to display many pattern objects on the right side of the window. Drag and drop one of these objects on the View pane of a File Browser window to change the background of all File Browser View panes. Drag and drop the Reset object to reset the background to its default color and pattern (usually white). The Colors button works the same way as the Patterns button. The Emblems button works the same way as the Emblems tab in the Side pane (page 278).

The **Edit⇒Preferences** selection displays the File Management Preferences window (Figure 8-6). This window has six tabs that control the appearance and behavior of File Browser windows.

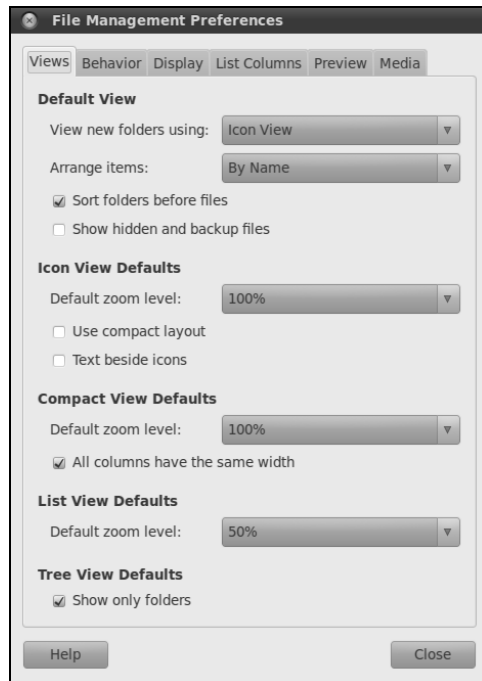


Figure 8-6 The File Management Preferences window, Views tab

The **Views** tab sets several defaults, including which view the File Browser displays (Icon, List, or Compact view), the arrangement of the objects, the default zoom level, and default settings for the Compact view.

Delete versus
Move to Trash

The **Behavior** tab controls how many clicks it takes to open an object and what Nautilus does when it opens an executable text object (script). For more confident users, this tab has an option that includes a Delete selection in addition to the Move to Trash selection on several menus. The Delete selection immediately removes the selected object instead of moving it to the **Trash** folder. This tab also holds the check box labeled **Open each folder in its own window** that is described in the next section.

The **Display** tab specifies which information Nautilus includes in object (icon) captions. The three drop-down lists specify the order in which Nautilus displays information as you increase the zoom level of the View pane. This tab also specifies the date format Nautilus uses.

The **List Columns** tab specifies which columns Nautilus displays, and in what order it displays them, in the View pane when you select **List View**.

The **Preview** tab controls when Nautilus displays or plays previews of files (Always, Local Files Only, Never).

The **Media** tab specifies which action Nautilus takes when you insert media such as a CD/DVD, or connect devices such as a USB flash drive, to the system.

View Click the **Main Toolbar**, **Side Pane**, **Location Bar**, and **Statusbar** selections in the View submenu to display or remove these elements from the window. The **Show Hidden Files** selection displays in the View pane those files with hidden filenames (page 204).

Go The Go selections display various folders in the View pane.

Bookmarks Bookmarks appear at the bottom of this menu and in the Side pane under Places. The Bookmarks selections are explained under “Places” on page 278.

Help The Help selections display local and online information about Nautilus.

optional

THE NAUTILUS SPATIAL VIEW

Nautilus gives you two ways to work with files: the traditional File Browser view described in the previous section and the innovative Spatial view shown in Figure 8-7. By default, Ubuntu displays the Browser view.

The Nautilus Spatial (as in “having the nature of space”) view has many powerful features but may take some getting used to. It always provides one window per folder. By default, when you open a folder, Nautilus displays a new window.

Turn on the Spatial view by selecting **File Browser menubar: Edit⇨Preferences**. Then click the **Behavior** tab in the File Management Preferences window and put a tick in the check box labeled **Open each folder in its own window**, click **Close**, and close the File Browser window. Next time you open a File Browser window, it will display a Spatial view.

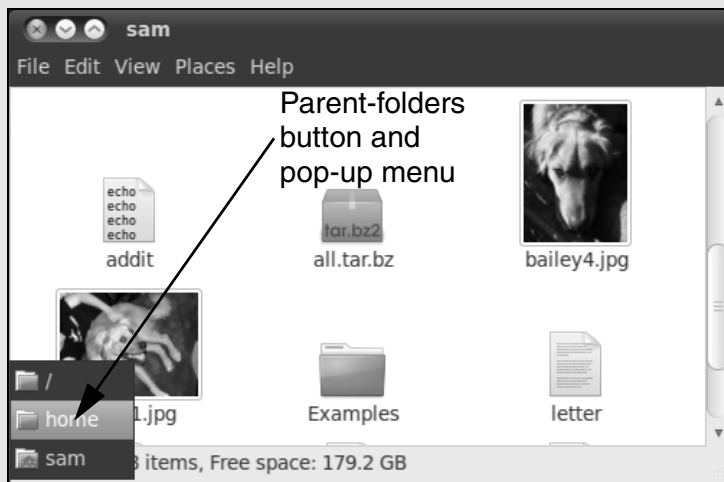


Figure 8-7 The Nautilus Spatial view

To open a Spatial view of your home directory, Select **Main menu: Home Folder** and experiment as you read this section. If you double-click the Desktop icon in the Spatial view, Nautilus opens a new window that displays the **Desktop** folder.

You can turn off the Nautilus Spatial view

tip To turn off the Nautilus Spatial view, open a File Browser window. From the menubar, open the File Management Preferences window by selecting **Edit⇒Preferences**. Click the **Behavior** tab in this window and remove the tick from the check box labeled **Open each folder its own window**.

A Spatial view can display icons, a list of filenames, or a compact view. To select your preferred format, click **View** on the menubar and choose **Icons**, **List**, or **Compact**. To create files to experiment with, right-click in the window (not on an icon) to display the Nautilus context menu and select **Create Folder** or **Create Document**.

Use SHIFT to close the current window as you open another window

tip If you hold the **SHIFT** key down when you double-click to open a new window, Nautilus closes the current window as it opens the new one. This behavior may be more familiar and can help keep the desktop from becoming cluttered. If you do not want to use the keyboard, you can achieve the same result by double-clicking the middle mouse button.

Window memory Move the window by dragging the titlebar. The Spatial view has *window memory*—that is, the next time you open that folder, Nautilus opens it at the same size and in the same location. Even the scrollbar will be in the same position.

Parent-folders button The key to closing the current window and returning to the window of the parent directory is the **Parent-folders** button (Figure 8-7). Click this button to display the Parent-folders pop-up menu. Select the directory you want to open from this menu. Nautilus then displays in a Spatial view the directory you specified.

From a Spatial view, you can open a folder in a traditional view by right-clicking the folder and selecting **Browse Folder**.

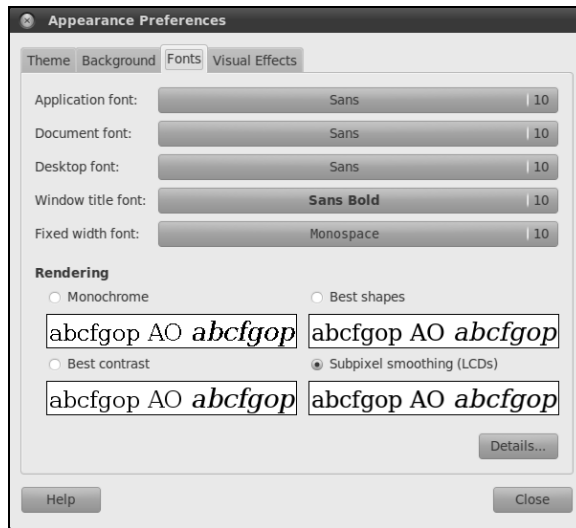


Figure 8-8 The Appearance Preferences window, Fonts tab

GNOME UTILITIES

GNOME comes with numerous utilities that can make your work with the desktop easier and more productive. This section covers several tools that are integral to the use of GNOME.

FONT PREFERENCES

The Fonts tab of the Appearance Preferences window (Figure 8-8) enables you to change the font GNOME uses for applications, documents, the desktop, window titles, and terminal emulators (fixed width). To display this window, select **Main menu: System**⇒**Preferences**⇒**Appearance** or enter `gnome-appearance-properties` on a command line. Click the **Fonts** tab. Click one of the five font bars in the upper part of the window to display the Pick a Font window (discussed next).

Examine the four sample boxes in the lower part of the window and select the one in which the letters look the best. Subpixel smoothing is usually best for LCD monitors. Click **Details** to refine the font rendering further, again picking the box in each section in which the letters look the best.

PICK A FONT WINDOW

The Pick a Font window (Figure 8-9) appears when you need to choose a font; see the previous section. From this window you can select a font family, a style, and a size. A preview of your choice appears in the Preview frame in the lower part of the window. Click **OK** when you are satisfied with your choice.

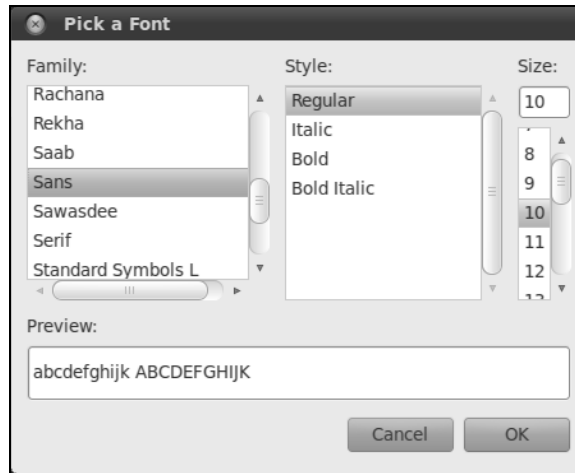


Figure 8-9 The Pick a Font window

PICK A COLOR WINDOW

The Pick a Color window (Figure 8-10) appears when you need to specify a color, such as when you specify a solid color for the desktop background (page 114) or a panel. To specify a color for a panel, right-click the panel to display its context menu, select **Properties**, click the **Background** tab, click the radio button labeled **Solid color**, and click within the box labeled **Color**. GNOME displays the Pick a Color window.

When the Pick a Color window opens, the bar below the color circle displays the current color. Click the desired color on the color ring, and click/drag the lightness of that color in the triangle. As you change the color, the right end of the bar below the color circle previews the color you are selecting, while the left end continues to display the current color. You can also use the eyedropper to pick up a color from the workspace: Click the eyedropper, and then click the resulting eyedropper mouse pointer on the color you want to select. The color you choose appears in the bar. Click **OK** when you are satisfied with the color you have specified.

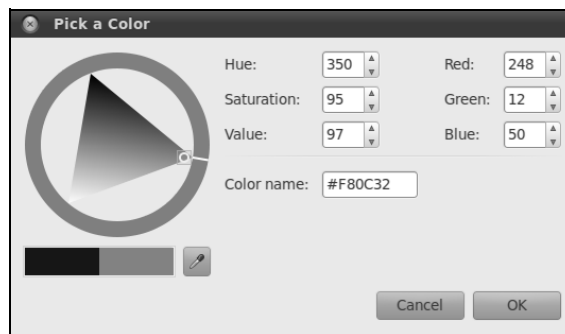


Figure 8-10 The Pick a Color window

RUN APPLICATION WINDOW

The Run Application window (Figure 4-4, page 103) enables you to run a program as though you had initiated it from a command line. To display the Run Application window, press **ALT-F2**. Enter a command in the text box. As soon as GNOME can uniquely identify the command you are entering, it completes the command and may display an object that identifies the application. Keep typing if the displayed command is not the one you want to run. Otherwise, press **RETURN** to run the command or **TAB** to accept the command in the text box. You can then continue entering information in the window. Click **Run with file** to specify a file to use as an argument to the command in the text box. Put a tick in the check box labeled **Run in terminal** to run a textual application, such as `vim.tiny`, in a terminal emulator window.

SEARCHING FOR FILES

The Search for Files window (Figure 8-11) can help you find files whose locations or names you do not know or have forgotten. To open this window, select **Main menu: Places** ⇒ **Search for Files** or enter `gnome-search-tool` on a command line from a terminal emulator or Run Application window (**ALT-F2**). To search by filename or partial filename, enter the (partial) filename in the combo box labeled **Name contains** and then select the folder you want to search in from the drop-down list labeled **Look in folder**. When GNOME searches in a folder, it searches subfolders to any level (it searches the directory hierarchy). To search all directories in all mounted filesystems, select **File System** from the drop-down list labeled **Look in folder**. Select **Other** to search a folder not included in the drop-down list; GNOME opens the Browse/Save window (page 110). Once you have entered the search criteria, click **Find**. GNOME displays the list of files matching the criteria in the list box labeled **Search results**. Double-click a file in this list box to open it.

To refine the search, you can enter more search criteria. Click the plus sign to the left of **Select more options** to expand the window and display more search criteria.

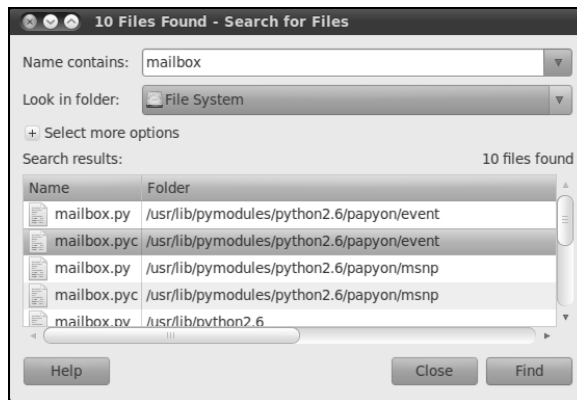


Figure 8-11 The Search for Files window

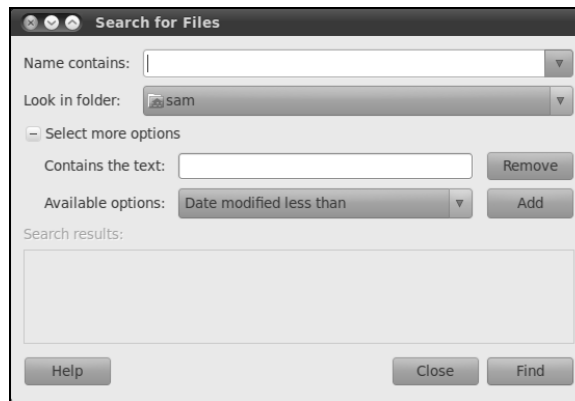


Figure 8-12 The Search for Files window with Select more options expanded

GNOME initially displays one search criterion and a line for adding another criterion as shown in Figure 8-12. With this part of the window expanded, GNOME incorporates all visible search criteria when you click **Find**.

The first line below **Select more options** holds a text box labeled **Contains the text**. If nothing is entered in this text box, the search matches all files. You can leave this text box as is or remove the line by clicking **Remove** at the right end of the line. To search for a file that contains a specific string of characters (text), enter the string in this text box.

To add search criteria, make a selection from the list box labeled **Available options** and click **Add** to the right of the drop-down list. To remove criteria, click **Remove** at the right end of the line that holds the criterion you want to remove.

To select files that were modified fewer than a specified number of days ago, select **Date modified less than** from the drop-down list labeled **Available options** and click **Add**. The Search for Files window adds a line with a spin box labeled **Date modified less than**. With this spin box showing **0** (zero), as it does initially, no file matches the search criteria. Change this number as desired and click **Find** to begin the search.

GNOME TERMINAL EMULATOR/SHELL

The GNOME terminal emulator displays a window that mimics a character-based terminal (page 125). To display a terminal emulator window, select **Main menu: Applications**⇒**Accessories**⇒**Terminal** or enter **gnome-terminal** on a command line or from a Run Application window (ALT-F2). When the GNOME terminal emulator is already displayed, select **Terminal menubar: File**⇒**Open Terminal** or right-click within the Terminal window and select **Open Terminal** to display a new terminal emulator window.

To open an additional terminal session within the same Terminal window, right-click the window and select **Open Tab** from the context menu or select **Terminal**

menubar: File⇒Open Tab. A row of tabs appears below the menubar as `gnome-terminal` opens another terminal session on top of the existing one. Add as many terminal sessions as you like; click the tabs to switch between sessions.

GNOME terminal emulator shortcuts

tip While using the GNOME terminal emulator, `CONTROL-SHIFT-N` opens a new window and `CONTROL-SHIFT-T` opens a new tab. New windows and tabs open to the working directory. In addition, you can use `CONTROL-PAGE UP` and `CONTROL-PAGE DOWN` to switch between tabs.

A session you add from the context menu uses the same profile as the session you open it from. When you use the menubar to open a session, GNOME gives you a choice of profiles, if more than one is available. You can add and modify profiles, including the Default profile, by selecting **Terminal menubar: Edit⇒Profiles**. Highlight the profile you want to modify or click **New** to design a new profile.

CHAPTER SUMMARY

The X Window System GUI is portable and flexible and makes it easy to write applications that work on many different types of systems without having to know low-level details for the individual systems. This GUI can operate in a networked environment, allowing a user to run a program on a remote system and send the results to a local display. The client/server concept is integral to the operation of the X Window System, in which the X server is responsible for fulfilling requests made of X Window System applications or clients. Hundreds of clients are available that can run under X. Programmers can also write their own clients, using tools such as the GTK+ and GTK+2 GNOME libraries to write GNOME programs and the Qt and KDE libraries to write KDE programs.

The window managers, and virtually all X applications, are designed to help users tailor their work environments in simple or complex ways. You can designate applications that start automatically, set such attributes as colors and fonts, and even alter the way keyboard strokes and mouse clicks are interpreted.

Built on top of the X Window System, the GNOME desktop manager can be used as is or customized to better suit your needs. It is a graphical user interface to system services (commands), the filesystem, applications, and more. Although not part of GNOME, the Metacity and Compiz window managers work closely with GNOME and are the default window managers for GNOME under Ubuntu. A window manager controls all aspects of the windows, including placement, decoration, grouping, minimizing and maximizing, sizing, and moving.

The Nautilus File Browser window is a critical part of GNOME; the desktop is a modified File Browser window. The File Browser View pane displays icons or a list of filenames you can work with. The Side pane, which can display six types of information, augments the information Nautilus displays in the View pane.

GNOME also provides many graphical utilities you can use to customize and work with the desktop. It supports MIME types; thus, when you double-click an object, GNOME generally knows which tool to use to display the data represented by the object. In sum, GNOME is a powerful desktop manager that can make your job both easier and more fun.

EXERCISES

1. a. What is Nautilus?
 - b. List four things you can do with Nautilus.
 - c. How do you use Nautilus to search for a file?
2. What is a terminal emulator? What does it allow you to do from a GUI that you would not be able to do without one?
3. How would you search the entire filesystem for a file named **today.odt**?
4. a. List two ways you can open a file using Nautilus.
 - b. How does Nautilus “know” which program to use to open different types of files?
 - c. What are the three common Nautilus control bars? Which kinds of tools do you find on each?
 - d. Discuss the use of the Nautilus location bar in textual mode.

ADVANCED EXERCISES

5. Assume you are using a mouse with nine pointer buttons defined. How would you reverse the effects of using the mouse wheel?
6. a. How would you use Nautilus to connect to the FTP server at `ftp.ubuntu.com`?
 - b. Open the following folders: **ubuntu**, **dist**, and **lucid**. How would you copy the file named **Contents-i386.gz** to the desktop? What type of file is **Contents-i386.gz**?
 - c. How would you open the **Contents-i386.gz** file on the desktop? How would you open the **Contents-i386.gz** file on the FTP server? Which file opens more quickly? Why? Which file can you modify?
7. Discuss the client/server environment set up by the X Window System. How does the X server work? List three X clients. Where is the client and where is the server when you log in on a local system? What is an advantage of this setup?

8. Run `xwininfo` from a terminal emulator window and answer these questions:
 - a. What does `xwininfo` do?
 - b. What does `xwininfo` give as the name of the window you clicked? Does that agree with the name in the window's titlebar?
 - c. What is the size of the window? In which units does `xwininfo` display this size? What is the depth of a window?
 - d. How can you get `xwininfo` to display the same information without having to click the window?
9. Find and install `xeyes` (not `tuxeyes`). Write an `xeyes` command to display a window that is 600 pixels wide and 400 pixels tall, is located 200 pixels from the right edge of the screen and 300 pixels from the top of the screen, and contains orange eyes outlined in blue with red pupils. (*Hint*: Refer to the `xeyes` man page.)