Navigation

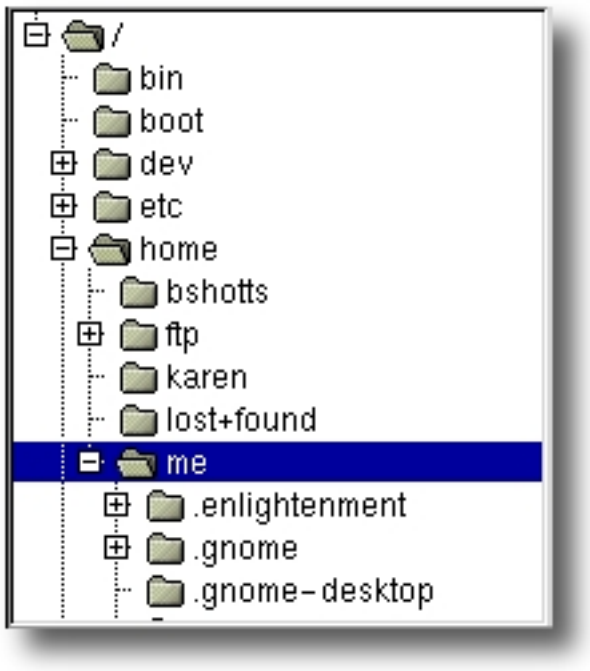
In this lesson, I will introduce your first three commands: **`pwd`** (print working directory), **`cd`** (change directory), and **`ls`** (list files and directories).

If you have not worked with a command line interface before, you will need to pay close attention to this lesson, since the concepts will take some getting used to.

File System Organization

Like that legacy operating system, the files on a Linux system are arranged in what is called a *hierarchical directory structure*. This means that they are organized in a tree-like pattern of *directories* (called folders in other systems), which may contain files and other directories. The first directory in the file system is called the *root directory*. The root directory contains files and subdirectories, which contain more files and subdirectories and so on and so on.

Most graphical environments today include a file manager program to view and manipulate the contents of the file system. Often you will see the file system represented like this:



One important difference between the legacy operating system and Unix-like operating systems such as Linux is that Linux does not employ the concept of drive letters. While drive letters split the file system into a series of different trees (one for each drive), Linux always has a single tree. Different storage devices may contain different branches of the tree, but there is always a single tree.

pwd

Since a command line interface cannot provide graphic pictures of the file system structure, it must have a different way of representing it. Think of the file system tree as a maze, and you are standing in it. At any given moment, you are located in a single directory. Inside that directory, you can see its files and the pathway to its *parent directory* and the pathways to the subdirectories of the directory in which you are standing.

The directory you are standing in is called the *working directory*. To find the name of the working directory, use the **pwd** command.

```
[me@linuxbox me]$ pwd
/home/me
```

When you first log on to a Linux system, the working directory is set to your *home directory*. This is where you put your files. On most systems, your home directory will be called /home/your_user_name, but it can be anything according to the whims of the system administrator.

To list the files in the working directory, use the **ls** command.

```
[me@linuxbox me]$ ls
Desktop     Xrootenv.0    linuxcmd
GNUstep     bin           nedit.rpm
GUILG00.GZ  hitni123.jpg  nsmail
```

I will come back to **ls** in the next lesson. There are a lot of fun things you can do with it, but I have to talk about pathnames and directories a bit first.

cd

To change your working directory (where you are standing in the maze) you use the **cd** command. To do this, type **cd** followed by the *pathname* of the desired working directory. A pathname is the route you take along the branches of the tree to get to the directory you want. Pathnames can be specified in one of two different ways; *absolute pathnames* or *relative pathnames*. Let's look with absolute pathnames first.

An absolute pathname begins with the root directory and follows the tree branch by branch until the path to the desired directory or file is completed. For example, there is a directory on your system in which most programs are installed. The pathname of the directory is /usr/bin. This means from the root directory (represented by the leading slash in the pathname) there is a directory called "usr" which contains a directory called "bin".

Let's try this out:

```
[me@linuxbox me]$ cd /usr/bin
[me@linuxbox bin]$ pwd
/usr/bin
[me@linuxbox bin]$ ls
[                       lwp-request
2to3                    lwp-rget
2to3-2.6                lxterm
a2p                     lz
aalib-config            lzcat
aconnect                lzma
acpi_fakekey            lzmadec
acpi_listen             lzmainfo
add-apt-repository      m17n-db
addpart                 magnifier

and many more...
```

Now we can see that we have changed the current working directory to /usr/bin and that it is full of files. Notice how your prompt has changed? As a convenience, it is usually set up to display the name of the working directory.

Where an absolute pathname starts from the root directory and leads to its destination, a relative pathname starts from the working directory. To do this, it uses a couple of special notations to represent relative positions in the file system tree. These special notations are "." (dot) and ".." (dot dot).

The "." notation refers to the working directory itself and the ".." notation refers to the working directory's parent directory. Here is how it works. Let's change the working directory to /usr/bin again:

```
[me@linuxbox me]$ cd /usr/bin
[me@linuxbox bin]$ pwd
/usr/bin
```

O.K., now let's say that we wanted to change the working directory to the parent of /usr/bin which is /usr. We could do that two different ways. First, with an absolute pathname:

```
[me@linuxbox bin]$ cd /usr
[me@linuxbox usr]$ pwd
/usr
```

Or, with a relative pathname:

```
[me@linuxbox bin]$ cd ..
[me@linuxbox usr]$ pwd
/usr
```

Two different methods with identical results. Which one should you use? The one that requires the least typing!

Likewise, we can change the working directory from /usr to /usr/bin in two different ways. First using an absolute pathname:

```
[me@linuxbox usr]$ cd /usr/bin
[me@linuxbox bin]$ pwd
/usr/bin
```

Or, with a relative pathname:

```
[me@linuxbox usr]$ cd ./bin
[me@linuxbox bin]$ pwd
/usr/bin
```

Now, there is something important that I must point out here. In almost all cases, you can omit the "./". It is implied. Typing:

```
[me@linuxbox usr]$ cd bin
```

would do the same thing. In general, if you do not specify a pathname to something, the working directory will be assumed. There is one important exception to this, but we won't get to that for a while.

A Few Shortcuts

If you type **cd** followed by nothing, **cd** will change the working directory to your home directory.

A related shortcut is to type **cd *~user_name***. In this case, **cd** will change the working directory to the home directory of the specified user.

Typing **cd -** changes the working directory to the previous one.

Important facts about file names

1. File names that begin with a period character are hidden. This only means that **`ls`** will not list them unless you say **`ls -a`**. When your account was created, several hidden files were placed in your home directory to configure things for your account. Later on we will take a closer look at some of these files to see how you can customize your *environment*. In addition, some applications will place their configuration and settings files in your home directory as hidden files.

2. File names in Linux, like Unix, are case sensitive. The file names "File1" and "file1" refer to different files.

3. Linux has no concept of a "file extension" like legacy operating systems. You may name files any way you like. However, while Linux itself does not care about file extensions, many application programs do.

4. Though Linux supports long file names which may contain embedded spaces and punctuation characters, limit the punctuation characters to period, dash, and underscore. **Most importantly, do not embed spaces in file names.** If you want to represent spaces between words in a file name, use underscore characters. You will thank yourself later.