Navigation

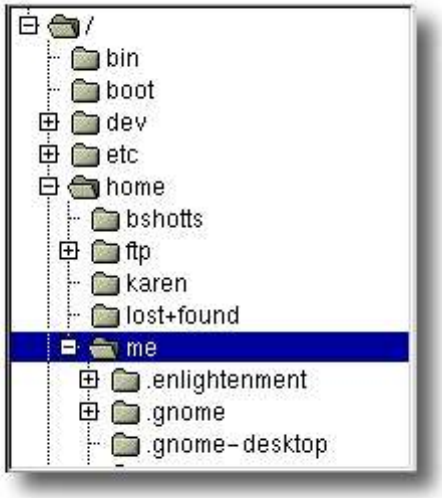
In this lesson, we will introduce our first three commands: **pwd** (print working directory), **cd** (change directory), and **ls** (list files and directories).

Those new to the command line will need to pay close attention to this lesson since the concepts will take some getting used to.

File System Organization

Like Windows, 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 *subdirectories*. 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 include a file manager program used to view and manipulate the contents of the file system. Often we will see the file system



represented like this:

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

pwd

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

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

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

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

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

```
[me@linuxbox me]$ ls
Desktop    Downloads         foo.txt  Pictures  Templates
Documents  examples.desktop  Music    Public    Videos
```

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

cd

To change the working directory (where we are standing in the maze) we use the **cd** command. To do this, we type **cd** followed by the *pathname* of the desired working directory. A pathname is the route we take along the branches of the tree to get to the directory we want. Pathnames can be specified 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
```

```
'['                                 mshortname
 2to3-2.7                           mshowfat
 411toppm                           mtools
 a2ps                               mtoolstest
 a2ps-lpr-wrapper                   mtr
 aa-enabled                         mtrace
 aa-exec                            mtr-packet
 aclocal                            mtvtoppm
 aclocal-1.15                       mtype
 aconnect                           mutter
 acpi_listen                        mxtar
 add-apt-repository                 mzip
 addpart                            namei

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 the shell 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 we 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 we must point out here. In most cases, we can omit the "./". It is implied. Typing:

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

would do the same thing. In general, if we 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 we type **cd** followed by nothing, **cd** will change the working directory to our 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 we 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 our *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 Windows 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.
