The UNIX File System

Computer file systems attempt to bring some sort of organization to the long-term data storage devices attached to computers: primarily I am talking about hard disks or CD-ROM drives. File systems provide a nomenclature by which contents of the file system can be unambiguously labeled, so that each file can be uniquely referred to by users or application programs. File systems also provide ways to create, read, update, and delete files and file system organizational structures.

The UNIX file system for a given computer running UNIX has a single directory hierarchy in what is known as a *tree structure*. Each directory can contain files and other (sub)directories. The top-most directory is called the *root directory*, and is notated with a single forward slash **/** . The analogy made by the reference to a tree structure works like how the main trunk of a tree (the file system root) can be branched (by creating subdirectories), and each of those branches can have branches. Eventually, there are leaves on branches that correspond to files in this analogy. Windows refers to directories as folders, and develops a similar file and directory hierarchy on storage devices it manages.

Unlike Windows, where each drive is designated with a drive letter, UNIX incorporates drives and drive partitions into its directory hierarchy through *mount points*, locations within the file system hierarchy which appear as directories.

Home Directory

As part of the command line user interface provided by a shell, the shell maintains the notion of a current or working directory. When you log in to a UNIX box, the directory that becomes your current directory is generally the directory known as your home directory. Each user has his/her own home directory. On our machine, user home directories are in the /usr/home directory.

Try typing the following at the command prompt, and hit Enter:

```
% pwd
```

You should see something like:

```
% pwd

/usr/home/btv01

%
```

This shows you the name of your home directory, which is **/usr/home/** followed by your user name. The **pwd** command stands for *print working directory*.

What's in your home directory? The command you use to find out what's in a directory is the **ls** command. Type the following at the command line now, followed by the Enter key.

```
% ls

%
```

You didn't see anything. That's OK. You are a new user and you haven't created any files yet. Create one as follows:

```
% touch firstfile
```

```
%
```

Now try the ls command:

```
% ls

firstfile

%
```

You actually have other files in your home directory; you just can't see them. Generally, programs will store user-specific configuration information in the user's home directory, or within a directory hierarchy based in the user's home directory. The names of these files generally start with a dot (.); the **ls** command does not show files whose names start with a dot unless you specifically request it to do so. Try the **ls** command with a special argument:

```
% ls -a

.    ..   .cshrc    .login    .profile       .shrc     firstfile

%
```

Most of these files configure the shells most commonly used on this machine, including the one you are using. The entry represented by the single dot indicates the current directory. The entry represented by the double dot indicates the directory to which this directory belongs (its *parent directory*).

You can move to the parent directory by:

```
% cd ..

%
```

The **cd** command stands for *change (current/working) directory*.

Now, find out what's the current directory (also known as the directory you are “in”) by:

```
% pwd

/usr/home

%
```

Go back to the previous directory by:

```
% cd -

/usr/home/btv##
```

This form of the cd command tells you where you are going. Verify by:

```
% pwd

/usr/home/btv##

%
```

Other File System Commands

For each of the following commands, type it in, press Enter, then type **ls** and press Enter. Note the results.

```
cp firstfile secondfile
cp secondfile thirdfile
mv thirdfile another
rm another
file firstfile
mkdir newdir
file newdir
mv newdir thisdir
rmdir thisdir
```

File and Directory Naming Conventions

You will have an easier time if you do not include spaces within names of files or directories.

You will also have an easier time for now if you just use the following characters in names of files and directories: digits, upper-case and lower-case letters, underscore, dot, comma.

File System and Shell Meta-characters

A metacharacter is a character that has a special meaning to the file system or the shell. It is generally a good idea to not use such characters in names of files or directories.

One useful metacharacter is the tilde (~). It can be used to refer to one's own home directory. One place you will see this is in certain URLs (Uniform Resource Locators, or the addresses of web pages). You might see the following as a URL:

http://www.somedomain.example/~johnsmith/

When you see this, you are working with a web site that is hosted on a machine running UNIX, and is presenting files somewhere within the home directory of a user on that machine.

Try the following commands:

```
% mkdir godir
% cd godir
% pwd
% cd ~
% pwd
% cd godir
% pwd
% cd
% pwd
```

Looking at File Contents

Try the following commands:

```
% cat .login
% head .login
% tail .login
% more .login
% less .login
```

Read the man pages for the above commands.

- the *cat* command outputs the contents of a file to the screen
- the *head* command outputs the first few lines of a file to the screen
- the *tail* command outputs the last few lines of a file to the screen
- the *more* command allows you to see the contents of a file on a page-by-page basis, rather than all at once as done by the *cat* command
- the *less* command is an enhanced form of the *more* command

Summary

You've learned:

- something about how the UNIX file system is organized
- that you have a home directory and where that is in the UNIX file system hierarchy
- that you have a working directory when you are at the command line and how to find out what it is
- how to look at the contents of a directory
- how to create, copy, rename, and delete files and directories
- file and directory naming conventions
- the tilde metacharacter
- commands that show you the types of and contents of files