Manipulating Files

This lesson will introduce you to the following commands:

- **cp** - copy files and directories
- **mv** - move or rename files and directories
- **rm** - remove files and directories
- **mkdir** - create directories

These four commands are among the most frequently used Linux commands. They are the basic commands for manipulating both files and directories.

Now, to be frank, some of the tasks performed by these commands are more easily done with a graphical file manager. With a file manager, you can drag and drop a file from one directory to another, cut and paste files, delete files, etc. So why use these old command line programs?

The answer is power and flexibility. While it is easy to perform simple file manipulations with a graphical file manager, complicated tasks can be easier with the command line programs. For example, how would you copy all the HTML files from one directory to another, but only copy files that did not exist in the destination directory or were newer than the versions in the destination directory? Pretty hard with with a file manager. Pretty easy with the command line:

```
[me@linuxbox me]$ cp -u *.html destination
```

Wildcards

Before I begin with our commands, I want to talk about a shell feature that makes these commands so powerful. Since the shell uses filenames so much, it provides special characters to help you rapidly specify groups of filenames. These special characters are called *wildcards*. Wildcards allow you to select filenames based on patterns of characters. The table below lists the wildcards and what they select:

Summary of wildcards and their meanings

Wildcard	***Meaning***
*	Matches any characters

?	Matches any single character
[characters]	Matches any character that is a member of the set *characters*. The set of characters can be expressed as a range of characters. (For example, [A-Z] represents all uppercase letters)
[!characters]	Matches any character that is not a member of the set *characters*

Using wildcards, it is possible to construct very sophisticated selection criteria for filenames. Here are some examples of patterns and what they match:

Examples of wildcard matching

Pattern	***Matches***
*	All filenames
*g**	All filenames that begin with the character "g"
b.txt*	All filenames that begin with the character "b" and end with the characters ".txt"
Data???	Any filename that begins with the characters "Data" followed by exactly 3 more characters
*[abc]**	Any filename that begins with "a" or "b" or "c" followed by any other characters
*[A-Z]**	Any filename that begins with an uppercase letter. This is an example of a range.
BACKUP.[0-9] [0-9][0-9]	Another example of ranges. This pattern matches any filename that begins with the characters "BACKUP." followed by exactly 3 numerals.
*[!a-z]**	Any filename that does not begin with a lowercase letter.

You can use wildcards with any command that accepts filename arguments.

cp

The **cp** program copies files and directories. In its simplest form, it copies a single file:

```
[me@linuxbox me]$ cp file1 file2
```

It can also be used to copy multiple files to a different directory:

```
[me@linuxbox me]$ cp file1 file2 file3 directory
```

Other useful examples of **cp** and its options include:

Examples of the cp command

Command	*Results*
cp file1 file2	Copies the contents of *file1* into *file2*. If *file2* does not exist, it is created; otherwise, *file2* is overwritten with the contents of *file1*.
cp -i file1 file2	Like above however, since the "-i" (interactive) option is specified, if *file2* exists, the user is prompted before it is overwritten with the contents of *file1*.
cp file1 dir1	Copy the contents of *file1* (into a file named *file1*) inside of directory *dir1*.
cp -R dir1 dir2	Copy the contents of the directory *dir1*. If directory *dir2* does not exist, it is created. Otherwise, it creates a directory named *dir1* within directory *dir2*.

mv

The **mv** command performs two different functions depending on how it is used. It will either move one or more files to a different directory, or it will rename a file or directory. To rename a file, it is used like this:

```
[me@linuxbox me]$ mv filename1 filename2
```

To move files to a different directory:

```
[me@linuxbox me]$ mv file1 file2 file3 directory
```

Examples of **mv** and its options include:

Examples of the mv command

Command	***Results***
mv file1 file2	If *file2* does not exist, then *file1* is renamed *file2*. If *file2* exists, its contents are replaced with the contents of *file1*.
mv -i file1 file2	Like above however, since the "-i" (interactive) option is specified, if *file2* exists, the user is prompted before it is overwritten with the contents of *file1*.
mv file1 file2 file3 dir1	The files *file1, file2, file3* are moved to directory *dir1*. *dir1* must exist or **mv** will exit with an error.
mv dir1 dir2	If *dir2* does not exist, then *dir1* is renamed *dir2*. If *dir2* exists, the directory *dir1* is created within directory *dir2*.

rm

The **rm** command deletes (removes) files and directories.

```
[me@linuxbox me]$ rm file
```

It can also be used to delete a directory:

```
[me@linuxbox me]$ rm -r directory
```

Examples of **rm** and its options include:

Examples of the rm command

Command	***Results***
rm file1 file2	Delete *file1* and *file2*.
rm -i file1 file2	Like above however, since the "-i" (interactive) option is specified, the user is prompted before each file is deleted.
rm -r dir1 dir2	Directories *dir1* and *dir2* are deleted along with all of their contents.

Be careful with rm!

Linux does not have an undelete command. Once you delete a file with **rm**, it's gone. You can inflict terrific damage on your system with **rm** if you are not careful, particularly with wildcards.

Before you use **rm** with wildcards, try this helpful trick: construct your command using **ls** instead. By doing this, you can see the effect of your wildcards before you delete files. After you have tested your command with **ls**, recall the command with the up-arrow key and then substitute **rm** for **ls** in the command.

mkdir

The **mkdir** command is used to create directories. To use it, you simply type:

```
[me@linuxbox me]$ mkdir directory
```
