An Introduction to Linux Basics

Authored by: ***ASPHostServer Administrator*** *[asphostserver@gmail.com]*
Saved From: *http://faq.asphosthelpdesk.com/article.php?id=205*

About Linux

Linux is a free, open-source operating system. All of ASPHostPortal's offered operating systems are Linux distributions.

Linux has been under active development since 1991. It has developed to become versatile and used all over the world, from web servers to cellphones.

ASPHostPortal offers Linux distributions on server instance's because Linux is free and easy to use.

However, newcomers to Linux may find it difficult to approach the structure of an unfamiliar operating system.

This guide gently introduces key terminal skills and equips newcomers to learn more about Linux.

The Terminal

For most of the time when you access a server, you'll be doing it through a terminal shell. The shell allows you to execute commands on the server instance.
All administrative tasks can be accomplished through the terminal. This includes file manipulation, package installation, and user management.

The terminal is interactive. You specify commands to run. The terminal outputs the results of those commands. Executing any command is done by typing it and pressing Enter.

Navigation

Linux filesystems are based on a directory tree. This means that you can create directories (or "folders") inside other directories, and files can exist in any directory.

To see what directory you are currently active in:

```
pwd
```

This stands for "print working directory", and will print the path to your current directory. The output can look similar to this:

```
/home/foo
```

This means that your current active directory is *foo*, which is inside *home*, which lives in the root directory, /.

To see other files and directories that exist in your current working directory:

```
ls
```

This will give you a list of names of files and directories. To navigate into a directory, use its name:

```
cd <name of directory>
```

This will change your new current working directory to the directory you specified. You can see this with `pwd`.

Additionally, you can specify `..` to change to the directory one level up in your path. To get back to your original directory:

```
cd ..
```

We can also create new directories in our current working directory. For example, to create a new directory called *bar*:

```
mkdir bar
```

Then we can `cd` into *bar* if we want. We can also delete *bar* if we no longer find it useful:

```
rm -d bar
```

`rm -d` will only delete empty directories.

File Manipulation

Files cannot be used with `cd` (it stands for "change directory").

Instead, we can view files. If we have a file *baz* in our current directory:

```
cat baz
```

This will print out the entire contents of *baz* to the terminal.

With long files, this is impractical and unreadable. To paginate the output:

```
less baz
```

This will also print the contents of *baz*, but one terminal page at a time, beginning at the start of the file.

Use the spacebar to advance a page, or the arrow keys to go up and down one line at a time. Press *q* to quit out of `less`.

To create a new file called *foobar*:

```
touch foobar
```

This creates an empty file with the name

foobar in your current working directory. The contents of this file are empty.

If we decide *foobar* isn't such a good name after all, we can rename *foobar* to *fizzbuzz*:

```
mv foobar fizzbuzz
```

`mv` stands for "move" and it can move a file or directory from one place to another.

By specifying the original file, we can "move" it to a new location in the current working directory, thereby renaming it.

It is also possible to copy a file to a new location. If we want to bring back *foobar*, but keep *fizzbuzz* too:

```
cp fizzbuzz foobar
```

Just as you guessed, `cp` is short for "copy". By copying *fizzbuzz* to a new file called *foobar*, we have replicated the original file in a new file with a different name.

However, what good is a file if it contains nothing? To edit files, a file editor is necessary.

There are many options for file editors, all created by professionals for daily use. Such editors include vim, emacs, nano, and pico.

nano is a perfectly suitable option for beginners. It is easy and simple to use, with no bells or whistles to confuse the average user.

To edit text into *foobar*:

```
nano foobar
```

This will open up a space where you can immediately start typing to edit *foobar*.

To save the written text, press `Ctrl-X` then *Y*. This returns you to the shell with a newly saved *foobar* file.

Now foobar has some text to view when using `cat` or `less`.

Finally, to delete the empty *fizzbuzz*:

```
rm fizzbuzz
```

Unlike directories, files are deleted whether they contain content or not.

The Filesystem Hierarchy Standard

Nearly all Linux distributions are compliant with a universal standard for filesystem directory structure.

The FHS defines clearly determined directores for different purposes.

The symbol / is used to indicate the root directory in the filesystem hierarchy defined by the FHS.

When a user logs into the shell, they'll be brought to their own user directory in /home.

The FHS defines /home as containing the home directories for regular users. (root has its own home directory in /root, also specified by the FHS.)

Because default, common-sense locations are provided for many different kinds of files, organisation of files for different purposes is simplified.

Permissions

On a system with multiple user accounts, determining who is allowed to interact with what files is important.

Linux supports unix-style filesystem permissions, which restricts who can read and write certain files.

Permissions are an expansive and profound topic, discussed in detail in our permissions article.

A Culture of Learning

So far, this guide only serves to teach the basics of toddling around in a Linux environment. But knowing your way around the Linux environment requires a mindset of study and illumination.

When you have a question about how to accomplish a task, there are several avenues of instruction you can turn to.

First and foremost, Google and DuckDuckGo are invaluable resources. Odds are that if you have a question, many others have already asked it and had the question answered.

Your immediate instinct should be to look for the answer through those search engines.

When your question has to do with any Linux command, the manual pages offer detailed and insightful documentation for nearly every single command.

To see the man-pages for any command:

```
man <command>
```

For instance, `man rm` displays the purpose of *rm*, how to use it, what options are available, examples of use, and more useful information.

Gaining the information you seek is an essential skill, which will sustain your Linux career for as long as you stay devoted to a time of learning.