2.4.1 Aliases

Click one of the buttons to take you to that part of the video.

Aliases 0:00-1:22

In this lesson, we're going to talk about aliases. An alias is, essentially, a shortcut to a command on your Linux system. For example, if you were to access the shell prompt on most Linux distributions and type the D-I_R command, dir, it will display a listing of files in the current directory. However, if you were to perform a search of the file system, you would find that there is no executable file anywhere on the system named dir. That's because dir is defined as an alias.

When you run the dir command with the shell prompt, the shell actually runs the command that dir is aliased to. Typically, you'll find that dir is aliased to the real command of ls -l. When you enter dir at the shell prompt, the shell does not run dir. What it actually does is run the ls -l command.

Likewise, most Linux distributions allow you to enter dot, dot (..) at the shell prompt. When you do, the shell takes you up one level in the file system. However, when you type '..', you're not really running a command named ..; the shell uses the .. alias to actually run the cd .. command because .. is aliased by default to that command, cd .. Understand that when your system boots, a series of default aliases are automatically created for you.

View Defined Aliases 1:23-1:47

You can view what they are by typing the alias command at the shell prompt. When you do, a list of all the currently defined aliases on your system are displayed. For example, you can see right here that dir runs the ls -l command. The .. alias runs the cd .. command. Also, notice that you could type —˜ll', and it would run the same command as the dir alias, ls -l.

Define new Aliases 1:48-3:05

Now, in addition to these default aliases, you can also define your own aliases. If you have a particular task that you need to run, and the command is really long, or, maybe, difficult to remember, you can define your own alias that might be a lot easier to type and allows you to get more done more efficiently. To define your own alias, you enter 'alias', and then you define a name for your alias. You can use whatever you want, whatever is easy for you to type. Then, you insert an equal sign (=) right here and then, in quotation marks (That's important. You have to put it in quotes), you put the actual command that you want the alias to run.

In this example, we want to be able to type the word details at the shell prompt, and when we do, we want it to run the ls -l command. We'll provide a long listing of all the files and folders within a particular directory. Be aware that if you use an alias name, right here, that already exists, then the new alias you define will overwrite the old one. Be very careful. Generally speaking, it's just a good idea to use unique alias names. With this alias defined, I can now go to the shell prompt, I can type 'details', and when I do, the ls -l command is going to be run for me.

Multiple Commands in a Single Alias 3:06-3:43

Another useful feature about aliases is the fact that you can actually include multiple commands within one single alias. All you have to do is separate the commands within the alias definition with a semicolon.

For example, you could create an alias for mounting optical disks. You could, in the first part of the alias, enter commands that mount the optical disk that's been inserted into an optical drive using the mount command. Then, after it's been mounted, you can display a long listing of all the files on that optical disk using the ls command. All you have to do is remember to separate the mount command and the ls command with semicolons.

Persistent Aliases 3:44-4:26

It's also important to note that the aliases that you define with the alias commands, such as we just looked at, are not persistent. For example, if you were to restart the system, then all the aliases that you manually defined at the shell prompt are going to be gone. Now, it's not the end of the world, because you can actually make your aliases persistent. All you have to do is add them to one of your shell configuration files. For example, if you want to make sure that these manually defined aliases are available to all the users on the system, you can add your alias commands to the /etc/profile shell configuration file. On the other hand, if you just want to make these alias commands persistent for your own individual user account, then you can add them to one of the hidden shell configuration files in your user's home directory.

Summary 4:27-4:38

That's how aliases work. In this lesson, we talked about what aliases are. We talked about how to view aliases that are currently defined on your system. We ended this lesson by talking about how to define new aliases on your system and to make them persistent across system reboots.