2.6.1 Shell Configuration Files

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

Shell Configuration Files 0:00-1:00

In this lesson, we're going to look at shell configuration files. Understand that you can customize your Bash shell environment by manipulating several different configuration files. We're going to look at those files in this lesson. Before we do that, be aware that these shell configuration files are not your typically configuration files. With a standard configuration file, we just have a standard text file. It has a parameter, usually an equal sign, and then a value we want to assign to that parameter.

A Bash shell configuration file doesn't work in this way. Instead, these files are actually scripts that are run whenever the shell session is started or whenever you log out from the shell. Because they're scripts, what you do is put in a series of commands that get executed when that script is run. However, which configuration files are used will actually depend upon two different things. First of all, what distribution you're using. Second of all, what type of shell you're using.

Login Shell 1:01-1:53

There are two categories of shells that you need to be familiar with. The first one is called a login shell. A login shell is in use if your system has booted into a text-based login screen and you used a text-based command line environment to interact with the system. For example, this is a common configuration on the Linux server systems.

An example of this is shown right here. In this system, I've booted into a text-based environment--a command line environment. I've logged in as the ksanders user, and I'm now interacting with the system from the command prompt. There's no graphical interface involved.

I do have to point out one caveat. Even if your Linux system is configured to boot into a GUI environment, into a graphical user interface environment, the login shell is still created when the system boots. You just don't see it.

Non-Login Shell 1:54-2:35

You can also start a non-login shell. A non-login shell is started from within a running system. In this example, I've booted into my graphical environment--as you can see here--and I've opened a terminal window within that environment. This is a non-login shell. It's a non-login shell because I didn't have to login to access the command prompt. I had already logged in to my graphical environment.

This distinction is important because the type of shell that you are using will dictate which configuration files are used to customize that particular shell environment. Also, be aware that different distributions also use different combinations of these files.

Non-Login Shell Configuration Files 2:36-5:54

The first shell configuration file you need to be familiar with is /etc/bashrc. On some distributions it's /etc/bash.bashrc. Now, bashrc or bash.bashrc are both configuration files that are used to configure non-login shells. They define functions and aliases that are used by all shell sessions for all users system-wide.

There's also another bashrc file, but it's a hidden file, and it's located in each individual user's home directory. That's what this little tilde mark (~) means right here. That's the user's home directory. Notice that because it has a period before the file name, it is a hidden file. On Linux, any file that you see that starts with a period is a hidden file. Now, .bashrc in the user's home directory is used to configure login shells as well. The difference is that it stores user-specific functions and aliases. /etc/bashrc contains functions and aliases for all users on the system. .bashrc in the user's home directory contains functions and aliases that are defined just for the individual user.

Now, there's one little caveat that I need to point out here, and that's the fact that even though bashrc is used to configure non-login shells--like a shell session that I open up in the graphical environment using a terminal program--the login shells on most distributions will actually use this file right here, as well. They'll call it right from the login shell configuration file. That's to make sure that the user-specific functions and aliases are defined no matter whether the user is logging in from a login shell or if the user has just opened a terminal window and is using a non-login shell. These two sets of files, here, are used for non-login shells.

Login Shell Configuration Files

The /etc/profile along with all of the files in the /etc/profile.d directory, on the other hand, are used for login shells. These files contain system-wide shell environment configuration parameters. The .bash_profile file in your user's home directory--again, it's a hidden file--is also used to configure login shells, but it stores user-specific shell preferences, whereas /etc/profile stores system-wide shell configuration parameters. That's applied to all users. /etc/profile applies to everybody. bash_profile only applies to one particular user.

On some distributions, it won't use the .bash_profile file in the user's home directory; it will instead use this file right here. It's just named .profile in the user's home directory. You'll need to look at the files in the individual user's home directory to see which one your distribution uses.

Now, some distributions will also use these two files here: .bash_login and .bash_logout. Both are hidden files in the individual user's home directory. As their name implies, they contain shell preferences that are applied either when the user logs in or when they log out. These two also apply only to login shells. Just remember that /etc/bashrc and .bashrc apply to non-login shells, and all the rest of these files apply to login shells.

Non-Login Script Order 5:55-6:17

In summary, if you're using a non-login shell--let's say you've logged into your graphical environment, and you just opened a terminal window--when this happens, the Bash shell first runs the /etc/bashrc file to configure the system-wide functions and aliases. Then it will run the .bashrc file inside the individual user's home directory to apply user-specific customizations.

Login Script Order 6:18-7:11

On the other hand, if you're using a login shell--say you've logged into a server system that doesn't have a graphical environment, and you logged in from a text-based prompt--then the Bash shell first runs the /etc/profile file and applies whatever configurations that are specified in there.

After that, though, the configuration file used depends upon which distribution you're using. For example, a Fedora system will use the .bashrc file in the user's home directory, as well as the .bash_profile file in the user's home directory, and the .bash_login file in the user's home directory. On the other hand, if you're doing this on an openSUSE system or an Ubuntu system, then the .bashrc file in the user's home directory and the .profile file in the user's home directory will be used instead.

Shell Configuration Files Precedence 7:12-8:11

Now, you need to be aware of the fact that if a login shell is being created, then the shell program searches for configuration files in the order shown here. First, it looks for the .bash_profile file. If it can't find that file, it will then look for the .bash_login file. If it can't find that file, then it looks for the .profile file. The important thing that you need to remember is the fact that the shell will use the first one of these files that it finds. As soon as it finds one of these files, it will ignore all of the rest. In my user's home directory, I could have all three of these files, and each one of them could have different parameters in them. But this would be the only one whose parameters would actually be applied because the shell will find it first, execute the script within this file, and then it stops. It does not continue looking through the remaining files.

Summary 8:12-8:24

That's it for this lesson. In this lesson, we discussed the files that are used to configure the Bash shell. We first talked about the difference between a login and a non-login shell session. Then we ended this lesson by reviewing all of the different configuration files that can be used to configure each type of shell.
