# 10.2.1 Process Management

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

Process Management 0:00-0:15

In this lesson, we're going to discuss managing running Linux processes. Specifically, we're going to look at running processes in the background and also switching processes between the background and the foreground.

---

Use subshell 0:16-1:52

Let's start by talking about how you run processes in the background. Recall that whenever you run any command at the shell prompt on a Linux system, a new subshell process is created. Then that process that you wanted to run is run within that subshell.

As soon as that process exits, then its associated subshell is destroyed as well. Here's the key thing that you need to remember.

During the time that that process remains running, the shell prompt of the parent shell itself disappears. And you can't do anything at the shell prompt, unless you were to open up a new terminal session. This happens because the process is running in what's called the foreground.

Now this behavior is even more apparent when you run a graphical application from the shell prompt. In this example, I've used the libreoffice command to launch the LibreOffice application.

Notice that while LibreOffice is running over here, the shell prompt is unavailable and is going to remain that way until I close this window by clicking on that X right there or clicking 'File' > 'Exit' over here. Only then can I enter additional commands at the shell prompt.

If I needed to enter another command without closing the LibreOffice application, I would have to open up a new terminal window over here and enter those commands there. Now this is the default behavior for any command that you enter at the shell prompt. It doesn't matter whether that command runs a text-based shell program or whether it runs a graphical program. The effect is the same.

---

Run in Background 1:53-3:26

Be aware that it is possible to run the program not in the foreground but in the background instead. If you do this, the program you launch will run normally, but control will be returned immediately to the shell session where you ran the command. Then you can use the shell to launch other programs or to perform other tasks.

The process for doing this is really quite easy. It's not that difficult to run a program in the background. In this example, I've run the same application that I ran in the previous example. I entered 'libreoffice' at the shell prompt, but notice that I added one character at the end of the command.

I added a space and then I appended an ampersand (&) character to the command. This is very important. This ampersand character tells the shell to run the program not in the foreground, but to instead run it in the background. By doing this, I still have the same application window, graphical application window that I had before, but I get my shell prompt back.

Now notice that when I ran this command with the ampersand that two values were displayed on the screen once the process was loaded into the background. The first value right here is the job ID and the second value is the process ID of the process being run to create this application.

Now this job ID number is very important because it helps you keep track of all of your background jobs. In fact, you can view a list of all the background jobs currently running on the system by entering just the simple jobs command at the shell prompt.

---

Switch Process Background and Foreground 3:27-6:26

As you can see in this example, I have just one background job running. Its job ID is 1, its status is running, and here's the name of the command that I used to create that background job.

Now just because a process was loaded into the background or to the foreground for that matter doesn't mean that it has to stay there. You can actually switch a process between the foreground and the background while it's still running.

Now what do you do if you have a process that's currently running in the foreground, and you decide, "Wait a minute, I actually want that running in the background."? One option would be to close the application and then load it again using the command that we just looked at--by appending an ampersand to the end of the command.

That is one way to do it. However, there is another, and I think better, way to do this.

Just because a process was started in the background--or in the foreground, it doesn't matter--doesn't mean it has to actually stay there without stopping.

You could actually switch a process between the foreground and the background while leaving it running. This is done using the two shell commands shown here. The first one is the fg command. This command will move a background process to the foreground.

Now in order to use the fg command, you do have to know the job ID number of that process that you want to move from the background to the foreground. Use the jobs command to find out what that is if you don't already know it.

Once you do, you just type fg and then the job ID number, and the process jumps from the background into the foreground. You can do just the opposite with the bg command. As its name implies, this command will take a process that's running in the foreground and move it into the background.

Now before you can do this, you do have to assign the foreground job a job ID number. Remember, when we loaded a process directly into the background, it was assigned a job ID number by default. If we have a foreground job, we've got to assign it one of those numbers before we can move it into the background. You do this by pressing the Ctrl z key sequence.

When you do this, you'll see the process stop, and a new job ID number will be assigned to that process. Once that's done, all you have to do is type bg followed by the job ID number that was just assigned, and it will move that process from the foreground to the background, and it'll continue running.

An example of how to do that is shown here. In this example, I ran the vi Editor to edit the zombie.c file. Notice that I did not include an ampersand right here.

Therefore, this process--the vi process--ran in the foreground. Then I decided "Wait a minute, I don't want it running in the foreground. I want it running in the background."

So I pressed Ctrl z. When I do, a job ID number was assigned, and the process was temporarily stopped. Here's the name of the command that was used to start the original process. Then I typed the bg command followed by the job number, which moved the process into the background, and it started running again.

---

Summary 6:27-6:38

That's it for this lesson. In this lesson we talked about managing running Linux processes. We first talked about how to run a process in the background. Then we looked at how you can move a process that's already running in the foreground to the background.

---