Whats a job in Linux

A job is a process that the shell manages. Each job is assigned a sequential job ID. Because a job is a process, each job has an associated PID. There are three types of job statuses:  
1. **Foreground**: When you enter a command in a terminal window, the command occupies that terminal window until it completes. This is a foreground job.  
2. **Background**: When you enter an ampersand (&) symbol at the end of a command line, the command runs without occupying the terminal window. The shell prompt is displayed immediately after you press Return. This is an example of a background job.  
3. **Stopped**: If you press Control + Z for a foreground job, or enter the stop command for a background job, the job stops. This job is called a stopped job.

Note: Except the Bourne shell, the other shells support job control.

Job Control Commands

Job control commands enable you to place jobs in the foreground or background, and to start or stop jobs. The table describes the job control commands.

|  |  |
| --- | --- |
| **Option** | **Description** |
| jobs | Lists all jobs |
| bg %n | Places the current or specified job in the background, where n is the job ID |
| fg %n | Brings the current or specified job into the foreground, where n is the job ID |
| Control-Z | Stops the foreground job and places it in the background as a stopped job |

**Note**: The job control commands enable you to run and manage multiple jobs within a shell. However, you can use the job control commands only in the shell where the job was initiated.

Running a Job in the Background

To run a job in the background, you need to enter the command that you want to run, followed by an **ampersand (&)** symbol at the end of the command line. For example, run the sleep command in the background.

$ sleep 100 &

[1] 1302

$

The shell returns the job ID, in brackets, that it assigns to the command and the associated PID. With the job ID, you can use the job control commands to manage the job whereas the kernel uses PIDs to manage jobs.

When a background job is complete and you press Return, the shell displays a message indicating the job is done.

[1] + Done sleep 100 &

$

Managing the background jobs

You can use the **jobs** command to list the jobs that are currently running or suspended in the background.

$ jobs

[1]+ Running sleep 100 &

You can use the fg command to bring a background job to the foreground.

$ fg % 1

sleep 100

**Note**: The foreground job occupies the shell until the job is completed, suspended, or stopped and placed into the background.

You can use the ‘**Control+Z** keys and **bg** command to return a job to the background. The Control+Z keys suspend the job, and place it in the background as a stopped job. The bg command runs the job in the background. For example:  
**1. Using CTRL+Z**

$ sleep 100

^Z

[1]+ Stopped sleep 100

$ jobs

[1]+ Stopped sleep 100

**2. Using bg**

$ bg % 1

[1]+ sleep 100 &

$ jobs

[1]+ Running sleep 100 &

Directory stack manipulation

**The Directory Stack**

|  |  |  |
| --- | --- | --- |
| • [Directory Stack Builtins](https://www.gnu.org/software/bash/manual/html_node/Directory-Stack-Builtins.html#Directory-Stack-Builtins): |  | Bash builtin commands to manipulate the directory stack. |

The directory stack is a list of recently-visited directories. The pushd builtin adds directories to the stack as it changes the current directory, and the popd builtin removes specified directories from the stack and changes the current directory to the directory removed. The dirs builtin displays the contents of the directory stack. The current directory is always the "top" of the directory stack.

The contents of the directory stack are also visible as the value of the DIRSTACK shell variable.