Understanding Commands

When the user hits the Return key after typing a command at a shell prompt it adds a final invisible newline character. This denotes the end of the command and indicates to the shell that it should then attempt to interpret that command. The Bash interpreter first reads the command line as "standard input" (stdin) and splits it into separate words broken by spaces or tabs. Each of these words is known as a "token". The interpreter next examines the first token to see if it is one of the shell's "built-in" commands or an executable program located on the file system.

When the first token is recognized as a built-in shell command the interpreter executes that command; otherwise, it searches through the directories on a specified path to find a program of that name. The interpreter will then execute a recognized built-in command or recognized program and display any result in the Terminal as "standard output" (stdout). Where neither is found the interpreter will display an error message in the Terminal as "standard error" (stderr).

Each program can accept standard input and can produce standard output and standard error messages.

The Bash **type** command can be used to determine whether a token is recognized as a built-in shell command or the location of a recognized program, or to display a message if none can be found:

1. At a command prompt type **type clear** then hit **Return** to see the location of the **clear** program on the filesystem

2. Next, type the command **type exit** then hit **Return** to discover that **exit** is in fact a built-in shell command

3. Now, type **type nosuch** then hit **Return** to see this token cannot be found to match a built-in command or program name

You can also use the built-in command **hash** to see a list of your recently issued program commands and the number of times executed (hits).

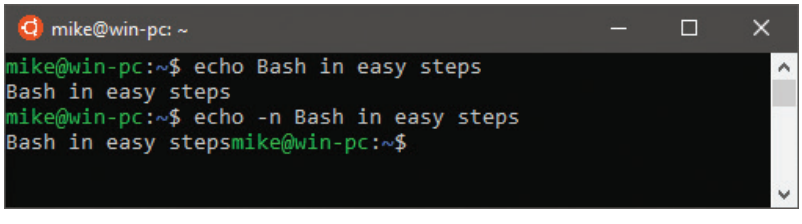
```
mike@win-pc: ~
mike@win-pc:~$ type clear
clear is /usr/bin/clear
mike@win-pc:~$ type exit
exit is a shell builtin
mike@win-pc:~$ type nosuch
-bash: type: nosuch: not found
mike@win-pc:~$
```

...cont'd

The Bash built-in **echo** command simply reads all following tokens from standard input then prints them as standard output – unless they are recognized as a command "option". Many built-in commands and programs accept one or more options that specify how they should be executed. Typically, an option consists of a dash followed by a letter. For example, the **echo** command accepts an **-n** option that denotes it should omit the newline character that it automatically prints after other output:

4 At a prompt, type **echo** followed by some text then hit **Return** to see that text printed with an added newline

5 Now, type **echo -n** followed by some text then hit **Return** to see that text printed without an added newline

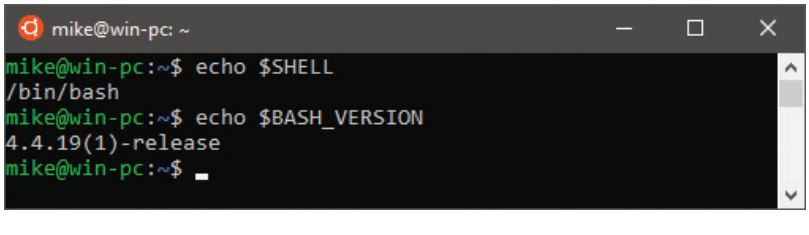


You will often want to suppress the automatic newline with **echo -n** when printing a request for user input.

In addition to the built-in shell commands the Bash shell also contains a number of built-in shell variables. These are named "containers" that each store a piece of information, and their names use all uppercase characters. To access the information stored within a variable its name must be prefixed with a **$** dollar sign:

6 At a prompt, enter **echo $SHELL** to see the location of the Bash interpreter program on the filesystem

7 Now, enter **echo $BASH_VERSION** to see the version number of the Bash shell interpreter



You can also use the command **echo $PATH** to discover which directories the Bash shell searches when you issue any command.