

Basics of Shell Scripting

github.com/grapheo12

Introduction

What is a Shell?

- Shell is a program that let's you run shell commands.
- A Shell is a command line interpreter, which runs your shell scripts.
- Shell **is not equal** to a Terminal.
- Examples:
 - tcsh
 - fish
 - Bourne - Again Shell (bash)
 - zsh etc.

What is a Terminal then?

- A terminal emulator is a graphical program that is used to run the shell.
- It emulates, although in a small window and with lots of add-ons, the tty interface.
- Examples:
 - GNOME Terminal
 - Konsole
 - urxvt
 - Alacritty
 - XTerm
 - Termite etc.

Wait! What is tty?

- tty (stands for Teletype) is text based interface to your operating system.
- On most Linux systems, you can use multiple teletype sessions and they can be accessed by hitting `Ctrl + Alt + F<1-12>` .
- What does a tty launch after you login into one? Yes, your default *shell*.

Pipes and redirections i

- Everything in Linux is a file. `stdin` , `stdout` and other streams are also (kinda) files.
- Shell commands (we'll see a few in a moment) generally spit there outputs to `stdout` and in some mode take input from `stdin` .
- What this means, is that we can redirect the output of one command to the input of the other without creating an intermediate file.
- To do so we use the pipe (`|`).
- For example, suppose we write out something using `echo` and want to print the number of lines in it using `wc` . The command for that will be:

Pipes and redirections ii

```
echo "Lorem ipsum dolor sit amet.\nSic mundus creatus est" | wc -l
```

- Operators for redirecting input/output from specific files / streams are:
 - `< inputfile` Redirects `stdin` to take input from a file.
 - `> outputfile` Redirects the output to a new file (Existing file is overwritten).
 - `>> outputfile` Appends the output to a file.
- An useful pattern is to redirect `stderr` to `/dev/null` (the black hole of Linux), so that it doesn't pollute your output: `2>/dev/null` .

Some Basic Commands

- This is used to list files.
- Usage: `ls -[Options] [path]`
- If a path is not given, current directory is assumed.
- Path can also contain wildcards. Example: `ls *.pdf` will list all the pdf files in the current directory.
- Options consists of one or a combination of character flags that invoke special functions:
 - `l` : List files with additional metadata
 - `a` : Show hidden files also. (Name begins with `.`)

- `h` : Show file sizes in MBs and GBs instead of bytes.
 - `t` : Sort by date modified.
 - ... and many more.
- Multiple options can be combined as: `ls -lah` .

Exercise: Open a shell and create a file with the list of all files (both hidden and visible) sorted by the date modified.

echo and printf i

- These are used to print stuff (Obvious Lol!)
- `printf` supports formatted output like C. `echo` doesn't.
- `echo` is more common in use than `printf` .
- Usage: `echo "some text"` or `printf "format string" "parameters"` .

echo and printf ii

Single and double quotes in bash

In bash, both single and double quotes are allowed. However there is subtle difference in behaviour. Inside double quoted string, you can use sub-commands enclosed by `$()` . This is not possible with single quotes. Run:

```
echo "$(ls /bin)"
```

and

```
echo '$(ls /bin)'
```

to see the difference.

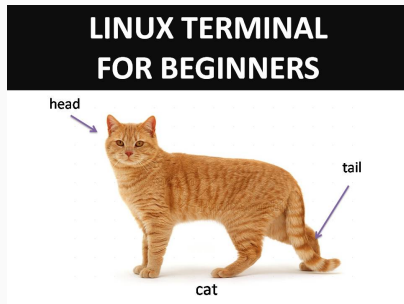


Figure 1: This is what these commands mean, literally!

- `cat` stands for `conCATenate` . `cat file1 file2 file3` will output all the 3 files combined in the given sequence.

cat, head and tail ii

- However, in practice, people use `cat` to print out the 1 full file only.
- `head` prints out the first few lines of a file and `tail` prints out the last few lines of a file.
- Both accept a parameter `-n<Number>` . This limits the output to `Number` number of lines.

Example: to get the first 15 lines of a line, run:

```
head -n15 file
```

grep i

- `grep` prints those lines in a given list of files that match a pattern.
- Usage: `grep pattern filename` .
- Another common usage is to pipe the output of some other command to `grep`. For example:

```
cat file | grep kharagpur
```

This will find all lines in a file that have the string “kharagpur” in it.

- `pattern` can be a Regular Expression too. For example:

```
whois google.com | grep [Cc]ountry
```

will fetch the whois record of Google.com using whois CLI (installed separately) and from that record, will find out all string which have either “country” or “Country” in it.

which

- Every command that you run in the shell actually is an executable located somewhere in your PATH (it is an environment variable, more on that later).
- To find out which particular executable is being run, `which` is used.

```
$ which echo  
/usr/bin/echo
```

Resolving virtualenvs

While working with multiple Python projects together, one might get confused as to what the current python binary is being used. At that time, running `which python` helps a lot.

cp, mv and mkdir

- `cp some/path/file some/other/path` copies `file` from `some/path` to `some/other/path` .
- To recursively copy a folder and all its files and subdirectories, we use `cp -r` .
- The main job of `mv` is to move files and folders from one directory to the other.
- Although `mv file newname` renames the file `file` to `newname` .
- Paths in `cp` and `mv` also support wildcards. For example, `cp yt-slides/*.pdf folder2/` copies only the pdf files.
- `mkdir` makes directories. Usage:
`mkdir existing/path/new_directory_to_make` . This creates a new directory `new_directory_to_make` under the existing path `existing/path` .
- However, if the parent directory doesn't exist yet, we can create the whole hierarchy

rm and rmdir

- `rmdir` removes empty directories.
- `rm` is a general command for removal of files and folders.
- To recursively delete, use the `-r` flag with `rm` .

Warning

NEVER RUN:

```
sudo rm -rf /
```

wc, sort, shuf, ...

These fall under the category of text manipulation programs.

- `wc` returns the newline, word and byte count for each of the files that are passed to it.
- We can get the individual newline, word or byte count by using `-l` , `-w` or `-c` flags respectively.
- `sort` sorts the lines of a document in lexicographical order. Although the ordering can be changed using appropriate flags.
- `sort -u` gives the unique lines in the document.
- `shuf` randomly selects a few lines from a file. The number of lines to take can be passed using `-n<Number>` flag.
- Other programs of this category are: `uniq` , `split` etc.

wget and curl

- These programs are used to fetch resources from the internet.
- `wget` , as the name suggests, performs only GET requests.
- By default, wget saves the output to a file in the current directory. However, this can be changed using the `-o` flag.
- cURL is a more generic tool. It can be used to perform arbitrary HTTP requests.

For example, sending a POST request to an URL through curl is as follows:

```
curl -X POST -H 'Content-type: application/json'  
-b '{"message": "Hello"}' http://url/endpoint
```

`-X` defines request method, `-H` defines headers, `-b` defines Request body.