Tutorial on head and tail (bash)

May 4, 2017

1 Table of Contents


```
In [1]: export LANGUAGE=en  # Be sure the commands output English text
```

2 1. Tutorial on `head` and `tail`

This short tutorial will show you how to use the linux command line tools `head` and `tail`, to respectively print the first lines and last lines of a file.

2.1 1.1 `head`

The first reflex is to print the help (option `--help`) :

```
In [2]: head --help
```

```
Usage: head [OPTION]... [FILE]...
Print the first 10 lines of each FILE to standard output.
With more than one FILE, precede each with a header giving the file name.
With no FILE, or when FILE is -, read standard input.

Mandatory arguments to long options are mandatory for short options too.
  -c, --bytes=[-]K         print the first K bytes of each file;
                             with the leading '-', print all but the last
                             K bytes of each file
  -n, --lines=[-]K         print the first K lines instead of the first 10;
                             with the leading '-', print all but the last
                             K lines of each file
  -q, --quiet, --silent    never print headers giving file names
  -v, --verbose            always print headers giving file names
      --help     display this help and exit
      --version  output version information and exit
```

```
K may have a multiplier suffix:
b 512, kB 1000, K 1024, MB 1000*1000, M 1024*1024,
GB 1000*1000*1000, G 1024*1024*1024, and so on for T, P, E, Z, Y.

GNU coreutils online help: <http://www.gnu.org/software/coreutils/>
Report head translation bugs to <http://translationproject.org/team/>
Full documentation at: <http://www.gnu.org/software/coreutils/head>
or available locally via: info '(coreutils) head invocation'
```

We will create a dummy file with random content, from the dictionnary (`/etc/dictionaries-common/words`):

```
In [3]: cat /etc/dictionaries-common/words > /tmp/file.txt
```

Now we can print the first 10 lines of this file `/tmp/file.txt`:

```
In [4]: head /tmp/file.txt
```

```
A
A's
AA's
AB's
ABM's
AC's
ACTH's
AI's
AIDS's
AM's
```

If we want the first 20 lines, we use the option `-n NUMBER`:

```
In [5]: head -n 20 /tmp/file.txt
```

```
A
A's
AA's
AB's
ABM's
AC's
ACTH's
AI's
AIDS's
AM's
AOL
```

```
AOL's
ASCII's
ASL's
ATM's
ATP's
AWOL's
AZ's
AZT's
Aachen
```

Let's count how many lines there is:

```
In [6]: wc -l /tmp/file.txt

99171 /tmp/file.txt
```

That's a lot! Imagine we want to see all the file except the last 100 lines, we can use `head` with a negative value for `NUMBER` in the `-n NUMBER` option:

```
In [7]: head -n 100 /tmp/file.txt | wc -l  # 100 lines, from 1st to 100th
        head -n -100 /tmp/file.txt | wc -l  # All lines but the last 100 lines, from 1st to 99

100
99071
```

2.2 1.2 `tail`

The other command, `tail`, works exactly like `head` except the lines are counted from the end:

```
In [8]: tail --help

Usage: tail [OPTION]... [FILE]...
Print the last 10 lines of each FILE to standard output.
With more than one FILE, precede each with a header giving the file name.
With no FILE, or when FILE is -, read standard input.

Mandatory arguments to long options are mandatory for short options too.
  -c, --bytes=K            output the last K bytes; or use -c +K to output
                             bytes starting with the Kth of each file
  -f, --follow[={name|descriptor}]
                           output appended data as the file grows;
                             an absent option argument means 'descriptor'
  -F                       same as --follow=name --retry
  -n, --lines=K            output the last K lines, instead of the last 10;
```

```
                           or use -n +K to output starting with the Kth
      --max-unchanged-stats=N
                         with --follow=name, reopen a FILE which has not
                           changed size after N (default 5) iterations
                           to see if it has been unlinked or renamed
                           (this is the usual case of rotated log files);
                           with inotify, this option is rarely useful
      --pid=PID          with -f, terminate after process ID, PID dies
  -q, --quiet, --silent  never output headers giving file names
      --retry            keep trying to open a file if it is inaccessible
  -s, --sleep-interval=N with -f, sleep for approximately N seconds
                           (default 1.0) between iterations;
                           with inotify and --pid=P, check process P at
                           least once every N seconds
  -v, --verbose          always output headers giving file names
      --help     display this help and exit
      --version  output version information and exit

If the first character of K (the number of bytes or lines) is a '+',
print beginning with the Kth item from the start of each file, otherwise,
print the last K items in the file.  K may have a multiplier suffix:
b 512, kB 1000, K 1024, MB 1000*1000, M 1024*1024,
GB 1000*1000*1000, G 1024*1024*1024, and so on for T, P, E, Z, Y.

With --follow (-f), tail defaults to following the file descriptor, which
means that even if a tail'ed file is renamed, tail will continue to track
its end.  This default behavior is not desirable when you really want to
track the actual name of the file, not the file descriptor (e.g., log
rotation).  Use --follow=name in that case.  That causes tail to track the
named file in a way that accommodates renaming, removal and creation.

GNU coreutils online help: <http://www.gnu.org/software/coreutils/>
Report tail translation bugs to <http://translationproject.org/team/>
Full documentation at: <http://www.gnu.org/software/coreutils/tail>
or available locally via: info '(coreutils) tail invocation'
```

```
In [9]: tail /tmp/file.txt  # Last 10 lines
```

```
élan's
émigré
émigré's
émigrés
épée
épée's
épées
étude
étude's
```

```
études
```

```
In [10]: tail -n 20 /tmp/file.txt  # Last 20 lines
```

```
zygote
zygote's
zygotes
Ångström
éclair
éclair's
éclairs
éclat
éclat's
élan
élan's
émigré
émigré's
émigrés
épée
épée's
épées
étude
étude's
études
```

The option `-n NUMBER` has the same behavior, except that it uses `-n +NUMBER` to ask for all lines but the first `NUMBER` (where `head` was asking `-n -NUMBER`) :

```
In [14]: tail -n 100 /tmp/file.txt | wc -l  # 100 lines, from 99071th to 99171th
         tail -n +101 /tmp/file.txt | wc -l  # All lines from line 101, from 101th to 99171th
```

```
100
99071
```

That's all for today, folks!