Part 2: Useful commands

Master your Command Line
(Before it masters you)

Tejas Sanap

12th July, 2019

Introduction

UNIX Philosophy

Focused on *modularity* & *reusability*.

It can be summarized as:

- Write programs that do one thing and do it well.
- Write programs to work together.
- Write programs to handle text streams, because that is a universal interface.

Basic Operations

All operations performed in the terminal can be categorized as:

- Search for text (in files).
  - `cat, head, tail, wc`
  - `grep`
- Search for files (in directories).
  - `find, locate`
- Manipulate text (in files).
  - `sed, awk, cut`
- Manipulate files (in directories).
  - `cp, scp, rm, mv, rsync`
  - `gzip, tar`
- Manipulate file permission and ownership.

GNU Coreutils

The GNU Core Utilities are the basic file, shell and text manipulation utilities of the GNU operating system.
They are expected to be present on every operating system.
Previously, the core utilities were implemented by the following pacakages:

1. `fileutils`
2. `shellutils`
3. `textutils`

In 2003, these three packages were combined into the current `coreutils` package.

Search

Text

cat, head, cd , wc

Utilities to view file content

Example

```
cat -A -n -s torrent-trackers
```

Example

```
head -n 10 torrent-trackers
```

Example

```
cd , cd .., cd ~, cd -
```

Example

```
wc torrent-trackers
```

wc - Output

```
465 233 9585 torrent-trackers
newline, wordcount, bytes, filename
```

`ls`

`ls` displays directory contents.

Useful `ls` options:

`--sort`	`-S`, `-t`, `-X` Size, time, extension
`--format`	`-1`, `-m`, `-l` Horizontal, commas, long
`-h`	human readable
`-g`	don't display file owner
`-G`	don't display file group
`-d`	list only directories
`-I`	Ignore files matching pattern
`--hide`	Hide files matching pattern (overriden by `-a`)

ls

Task

1. List all the directories in the folder `find`
2. List the last five files/folders to be modified

Example

```
$ ls
```

ls

Task

1. List all the directories in the folder `find`
2. List the last five files/folders to be modified

Example

```
$ ls -d *\
```

`ls`

Task

1. List all the directories in the folder `find`
2. List the last five files/folders to be modified

Example

```
$ ls -1t | head
```

grep

grep prints line that matches a certain pattern.

Syntax

```
grep OPTIONS PATTERN INPUT_FILE_NAMES
```

Example

```
$ grep --color=always "anime" torrent-tracker
udp://tc.animereactor.ru:8082/announce
udp://tc.animereactor.ru:8082/announce
```

grep

The exit status of `grep` when:

- **line is selected** is **0**.
- **no line is selected** is **1**.
- **an error occurs** is **2**.

Useful `grep` options:

`-i`	ignore case
`-v`	invert matches
`-c`	count no. of matching lines
`-n`	prefix each line with line number
`-l`	print name of the file and suppress all other output
`-H`	print filename for each match
`-o`	print only the matched parts of a line
`-s`	suppress error messages
`--color`	color the matching content
`-a`	accept binary input
`--label=LABEL`	display input actually coming from `stdin` as input from file `LABEL`

grep

Task

1. We have a tar file named `python_code.tar.gz`
2. We want to search for a function named `main`
3. But, without, extracting or decompressing the tar file

Example

```
$ tar -xf python_code.tar.gz
```

grep

Task

1. We have a tar file named `python_code.tar.gz`
2. We want to search for a function named `main`
3. But, without, extracting or decompressing the tar file

Example

```
$ tar -xzf python_code.tar.gz
```

grep

Task

1. We have a tar file named `python_code.tar.gz`
2. We want to search for a function named `main`
3. But, without, extracting or decompressing the tar file

Example

```
$ tar -xzf python_code.tar.gz --to-command='grep
main'
```

grep

Task

1. We have a tar file named `python_code.tar.gz`
2. We want to search for a function named `main`
3. But, without, extracting or decompressing the tar file

Example

```
$ tar -xzf python_code.tar.gz --to-command='grep -a
main'
```

grep

Task

1. We have a tar file named `python_code.tar.gz`
2. We want to search for a function named `main`
3. But, without, extracting or decompressing the tar file

Example

```
$ tar -xzf python_code.tar.gz --to-command='grep -a
-H main'
```

grep

Task

1. We have a tar file named `python_code.tar.gz`
2. We want to search for a function named `main`
3. But, without, extracting or decompressing the tar file

Example

```
$ tar -xzf python_code.tar.gz --to-command='grep -a
-H --label="$TAR_FILENAME" main'
```

grep

Task

1. We have a tar file named `python_code.tar.gz`
2. We want to search for a function named `main`
3. But, without, extracting or decompressing the tar file

Example

```
$ tar -xzf python_code.tar.gz --to-command='grep -a
-H --label="$TAR_FILENAME" -n main'
```

grep

Task

1. We have a tar file named `python_code.tar.gz`
2. We want to search for a function named `main`
3. But, without, extracting or decompressing the tar file

Example

```
$ tar -xzf python_code.tar.gz --to-command='grep -a
-H --label="$TAR_FILENAME" -c main'
```

grep

Task

1. We have a tar file named `python_code.tar.gz`
2. We want to search for a function named `main`
3. But, without, extracting or decompressing the tar file

Example

```
$ tar -xzf python_code.tar.gz --to-command='grep -a
-H --label="$TAR_FILENAME" -c -s main'
```

Files

find

`find` search for files in a directory hierarchy.

Syntax

```
find DIRECTORY EXPRESSION
```

find

`find` search for files in a directory hierarchy.

Syntax

```
find DIRECTORY TESTS ACTIONS
```

`find`

`find` search for files in a directory hierarchy.

Syntax

```
find DIRECTORY TESTS ACTIONS
```

Example

```
$ find . -name file1b1
```

`find`

`find` search for files in a directory hierarchy.

Syntax

```
find DIRECTORY TESTS ACTIONS
```

Example

```
$ find . -name file1b1
```

Useful global options:

`-maxdepth n`	Descend at most `n` levels
`-mindepth n`	Do not apply tests at levels less than `n`

`find`

Following TESTS are available:

Name	`-name, -iname, -path, -ipath`
Links	
Time	`-atime, -ctime, -mtime, -amin, -cmin, -mmin, -anewer, -cnewer, -mnewer, -newerXY, -used`
Size	`-size, -empty`
Type	`-type`
Owner	`-user, -group`
Mode Bits/ File Permissions	`-perm, -readable, -writable, -executable`
Contents	
Directories	
Filesystems	

find

-path

```
$ find . -path '*/dir4a'
./dir1/dir1a/dir2c/dir3a/dir4a
```

find

Task

Find files that were edited before:

1. 10 days.
2. 10 minutes.

find

Task

Find files that were edited before:

1. 10 days.
2. 10 minutes.

-newerXY

```
$ find . -newermt "Jul 11"
```

find

Task

Find files that were edited before:

1. 10 days.
2. 10 minutes.

-newerXY

```
$ find . -newermt "10:20"
```

find

-size

```
$ find . -size +5k $ find . -size -5k
```

find

Content

```
$ find . -name '*.[23]' | xargs grep -l anime
./dir1/dir1a/dir2c/dir3a/file4.2
./dir1/dir1b/file1b.3
```

`locate`

locate

Manipulate

Text

`sed`

sed

awk

awk

cut

cur

Files

scp

- download `znc.pem` from server to add to irssi client

rm, cp & mv

- Text globbing - Use latex compile files and stuff as examples
- Bash Pattern Matching

```
rm pre*.!(tex)
```

`gzip, tar`

Shell porn

fortune & cowsay

- Let's add some star trek quotes
- Cowthink and cowsay
- Add some bling with pony

Questions?

References

References

1. UnixPin
2. `man 7 regex`
3. `find`:
   3.1 Find History
   3.2 GNU Findutils - `info` –> Find