Command line Interface

Getting help from command line.

In Unix systems we can get help to understand any command in two different ways. First will be `commandName -h` and the second one is **man** command which is used in this way. `man commandName` . Let's see a couple of examples.

```
$ touch --h
usage:
touch [-A [-][[hh]mm]SS] [-acfhm] [-r file] [-t [[CC]YY]MMDDhhmm[.SS]] file ...
```

Let's show the man page for command touch. By typing this it will open the man page for the command.

```
$ man touch
```

Once you finish you can close it typing **q**.

Online Linux Terminal

If you are using a Windows system, you can use this Linux online terminal: https://cocalc.com/doc/terminal.html

Creating Files

In Unix systems we can use a single **>** to add in an empty file or overwrite the content in a file, or just use double **>>** in order to add instead to overwrite

Creating files with touch

```
$ touch script.sh
```

Creating empty file with echo

```
$ echo "" > script.sh
```

Creating file with echo

```
$ echo "This is a string content" > script.sh
```

Creating empty file with printf

```
$ printf "" > script.sh
```

Creating file with printf

```
$ printf "We can add what we need to include in the file here" > script.sh
```

Creating Files with editor vi

```
$ vi script.sh
```

Creating Files with editor vim

```
$ vim script.sh
```

Creating Files with editor nano

```
$ nano script.sh
```

All them will open given editor interface where you can add whatever you need to the file and save it.

Examining File content

```
$cat script.sh
We can add what we need to include in the file here
```

Adding lines to a file with cat command by adding

```
$ cat > file.txt
This text is going to be added to file.txt
```

Removing Files

To remove a file we userm command with the name of the file to be deleted

```
$ rm script.sh
```

Creating Directories

Creating Directory scripts (default mode is rwx for owner, rx group, rx others)

```
$ mkdir scritps
```

Listing directories

Basic listing

```
$ ls
README.md    bootstrap.sh images       script.sh    scripts      src          static
```

Listing directory elements in list mode

```
$ ls -l
-rw-r--r--  1 jose  staff  2576 Jun  2 11:21 README.md
-rw-r--r--  1 jose  staff    74 Jun  2 11:21 bootstrap.sh
drwxr-xr-x  2 jose  staff    64 Jun  2 11:19 images
-rw-r--r--  1 jose  staff    58 Jun  2 10:48 script.sh
drwxr-xr-x  2 jose  staff    64 Jun  2 10:37 scripts
drwxr-xr-x  2 jose  staff    64 Jun  2 11:18 src
drwxr-xr-x  2 jose  staff    64 Jun  2 11:19 static
```

Listing hidden directory elements in list mode

```
$ ls -la
drwxr-xr-x  11 jose  staff   352 Jun  2 11:21 .
drwxr-xr-x   3 jose  staff    96 Jun  2 10:18 ..
drwxr-xr-x  10 jose  staff   320 Jun  2 11:21 .git
drwxr-xr-x  15 jose  staff   480 Jun  2 11:21 .idea
-rw-r--r--   1 jose  staff  2940 Jun  2 11:21 README.md
-rw-r--r--   1 jose  staff    74 Jun  2 11:21 bootstrap.sh
drwxr-xr-x   2 jose  staff    64 Jun  2 11:19 images
-rw-r--r--   1 jose  staff    58 Jun  2 10:48 script.sh
drwxr-xr-x   2 jose  staff    64 Jun  2 10:37 scripts
drwxr-xr-x   2 jose  staff    64 Jun  2 11:18 src
drwxr-xr-x   2 jose  staff    64 Jun  2 11:19 static
```

Listing size of directory elements with a readable file size

```
$ ls -lh
-rw-r--r--  1 jose  staff   3.4K Jun  2 11:22 README.md
-rw-r--r--  1 jose  staff    74B Jun  2 11:21 bootstrap.sh
drwxr-xr-x  2 jose  staff    64B Jun  2 11:19 images
-rw-r--r--  1 jose  staff    58B Jun  2 10:48 script.sh
drwxr-xr-x  2 jose  staff    64B Jun  2 10:37 scripts
drwxr-xr-x  2 jose  staff    64B Jun  2 11:18 src
drwxr-xr-x  2 jose  staff    64B Jun  2 11:19 static
```

Listing recursively directory elements

```
$ ls -R
README.md    bootstrap.sh images       script.sh    scripts      src          static

./images:
assembler.png

./scripts:

./src:

./static:
main.css
```

Listing directory elements sorted by file size

```
$ ls -S
README.md    images       static       bootstrap.sh scripts      src          script.sh
```

Listing directory elements sorted by time and date

```
$ ls -t
README.md    static       images       bootstrap.sh src          script.sh    scripts
```

Listing directory elements sorted by extension name

```
$ ls -X
README.md
```

Listing directory elements in reverse order

```
$ ls -r
static       src          scripts      script.sh    images       bootstrap.sh README.md
```

Cleaning the terminal window

There are times when we need to clear the terminal screen. For that we use `clear` command

```
$ clear
```

Moving between directories

First thing we need to know.. Where we are just now? For that we use `pwd` command

```
$ pwd
/Users/jose/projects/AssemblerSchool/terminal-exercises-1
```

Moving between directories with cd command also known as change directory (cd)

```
$ cd scripts
$ pwd
/Users/jose/projects/AssemblerSchool/terminal-exercises-1/scripts
```

Let's move back! There's a shortcut to move up one directory level.

```
$ cd ..
$ pwd
/Users/jose/projects/AssemblerSchool/terminal-exercises-1
```

Let's move user home! There's a shortcut to move user homecd

```
$ cd
$ pwd
/Users/jose
```

Let's move to a given path/Users/jose/projects/AssemblerSchool

```
$ pwd
/Users/jose
$ cd /Users/jose/projects/AssemblerSchool
$ pwd
/Users/jose/projects/AssemblerSchool
```

Removing Directories

To remove the directory we also userm command with -rf options

```
$ rm -rf scripts
```

To remove all files and directories in given directory but not directory

```
$ rm -rf scripts/*
```

Moving things around

Moving file script.sh to scripts folder

```
$ mv script.sh scripts
```

Moving directory scripts to src folder

```
$ mv scripts src
ls -R
./src/scripts:
script.sh
```

Moving JS files to static folder. For this we use wildcard (*) character

```
$  mv *.js static/
```

Moving file with interactive prompt before overwrite. We answer y/n to proceed to overwrite or not the file

```
$ mv -i assembler.png images
overwrite images/assembler.png? (y/n [n])
y
```

Renaming things

We also can use mv command in order to rename a file or a directory

```
$ mv assembler.png assembler_logo.png
$ mv images user_images
```

Running an application from command line

We can run from command line an application. This can be done differently in Unix, Linux, Mac or Windows. As we need to know the full path of where the application is set. We are about to see **whych** command

We use which command in order to know the application path.

```
$ whereis vim
/usr/bin/vim
```

To open an application from terminal with Unix method we use sh -c and we pass the path of the application to the command

```
$ sh -c /usr/bin/vim
```

To open an application from terminal in Debian based systems like Ubuntu we just can type the application name

```
$ firefox http://www.assemblerschool.com
```

In Debian based systems we can also do this to open command window. Over a terminal press Alt+F2 it will show up the command window which is a little window there we can type application name

```
$
```

In MacOs systems we can use the command open with -a option which will run the application we type.

```
$ open -a Safari http://www.assemblerschool.com
```

In Windows systems we can use the command start

```
$ start microsoft-edge:http://www.assemblerschool.com
```