1. The Shell Command-Line utility

is a program that can be operated from the command line or terminal interface, without a graphical user interface (GUI). These utilities are designed to perform specific tasks or functions in a text-based environment and are common in Unix, Linux, macOS, and even Windows operating systems.

Command-line utilities accept commands and options (also known as flags or switches) as text input from the user. These commands tell the utility what action to perform, and the options modify how the command is executed. The output from these utilities is typically text that is printed to the terminal.

Examples

- `ls` : Lists the contents of a directory on Unix/Linux systems.
- `grep` : Searches for patterns within text. It is used to search text using complex patterns called regular expressions.
- `mysql` : A command-line tool for interacting with MySQL and MariaDB databases, allowing for the execution of SQL queries, database management, and more.
- `git` : A version control system that is used for tracking changes in source code during software development. It has a command-line utility for managing repositories.
- `curl` : A tool to transfer data from or to a server, using one of the supported protocols (HTTP, HTTPS, FTP, and more).
- `ssh` : Secure Shell is a network protocol that provides administrators with a secure way to access a remote computer.

Commands

- `~` : Tilde always expand to home directory
- `-` : if do `cd -`, will cd to the directory previous in
- `ls -l` : Command with Flag
- `mv <old place> <new place>` : can move a file to a different location & rename it (optional)
- `rm` remove file

What's in File System?

Computer has a bunch of built-in programs that comes with the machine, stored in file system
The shell has a way to determine where a program is located basically has a way to search for programs
It does this through invariant environment variable.

Environment variable

Things are set whenever you start your shell

e.g. where's the home directory / username / path variable `$PATH`
(all of the paths on the machine that the shell will search for programs)

```
(base) ishtar@iLouHdeMacBook-Air ~ % echo $PATH
/Users/ishtar/Downloads/apache-maven-3.9.6/bin:/Users/ishtar/opt/anaconda3/bin:/
Users/ishtar/opt/anaconda3/condabin:/opt/homebrew/bin:/opt/homebrew/sbin:/Librar
y/Frameworks/Python.framework/Versions/3.10/bin:/usr/local/bin:/System/Cryptexes
/App/usr/bin:/usr/bin:/bin:/usr/sbin:/sbin:/var/run/com.apple.security.cryptexd/
codex.system/bootstrap/usr/local/bin:/var/run/com.apple.security.cryptexd/codex.
system/bootstrap/usr/bin:/var/run/com.apple.security.cryptexd/codex.system/boots
trap/usr/appleinternal/bin:/usr/local/MacGPG2/bin:/Library/TeX/texbin
```

- a list
- colon separated

1. whenever you type the name of your program
2. the shell is gonna **search through the list of PATHs** on the machine
3. and it's gonna look in each directory for a program or a file whose name matches the command you try to run

PATH

Path are a way to name the location of a file on the computer
e.g.

```
(base) ishtar@iLouHdeMacBook-Air ~ % which ls
/bin/ls
```

- Starting at *the top of the file system* --> then look inside the directory called " `bin` ", then look for the file called " `ls` "
- On Linux or MacOS, everything lives under the root name space, all paths start with a slash or absolute paths

Absolute Path

Paths that fully determine the location of a file (the full path to that file)

Relative Path

Paths that is relative to where you currently are `pwd` -> present working directory

File Permission

e.g.

```
(base) ishtar@iLouHdeMacBook-Air Desktop % ls -l
total 672
drwxr-xr-x  10 ishtar  staff     320 Jan 26 11:38 Computer Science
-rw-------@  1 ishtar  staff   25948 Feb 14 08:54 Individual Supervisors.xlsx
```

`d` : directory

`-` : file

the rest : permission of the directory / file

- the first group(`rwx`) --> owner of the file --> `ishtar`
- the second group(`r-x`) --> group owner of the file --> `staff`
- the third group(`r-x`) --> everyone else
-

Directory Permission

`r` : read a directory

Are you allowed to see which files are inside the directory (`ls` ?)

`w` : write a directory

whether you are allowed to rename, create or remove files within that directory

`x` : execute a directory

search --> are you allowed to enter this directory (`cd` ?)

Streams

Chain programs, interact with files, have files operate in between programs

Every programs has 2 primary streams:

by default, input stream --> keyboard

output stream --> screen

Redirect the Streams

`<` : rewire the **input** for this program to be the contents of the file

`>` : rewire the **output** of the preceding program into this file

`>>` : append to the file

`|` : pipe --> take the **output** of the program to the left, and make the **input** of the program to the right

`grep` : Allow you to search in an input stream for a given pattern

`open <file>` : open a file

`cat` : prints the contents of the file

e.g. `cat hello.txt` , the shell is going to open `hello.txt` take its contents, set that to be the input of `cat` , and `cat` is going to print that to its output

e.g. `ls -l | head -n5 > ls.txt`

```
(base) ishtar@iLouHdeMacBook-Air Desktop % ls -l | head -n5 > ls.txt
(base) ishtar@iLouHdeMacBook-Air Desktop % cat ls.txt
total 352
drwxr-xr-x  10 ishtar  staff    320 Jan 26 11:38 Computer Science
-rw-------@  1 ishtar  staff  25948 Feb 14 08:54 Individual Supervisors.xlsx
drwxr-xr-x   3 ishtar  staff     96 Mar  4 13:10 JAVA-CW-2023
drwxr-xr-x  14 ishtar  staff    448 Mar  4 15:24 Java
```

Root User

sort of like the administrator, a super user with all the privileges

`sudo <command>`

`sudo` : do the following thing as a super user...

`sudo su`

`su` : gets you a shell as the super user

switch into a root terminal

```
(base) ishtar@iLouHdeMacBook-Air Desktop % sudo su
Password:
sh-3.2#
```

`echo 1060 | sudo tee brightness`

`tee`

takes an input and writes to a **file** but also to **standard out**