Terminal

→ The Terminal = A text based input and output environment

- Command Line = Any interface that is used by entering textual commands (generally Windows centric)

- Terminal = This is a type of command line (generally Mac centric)

- Console = A command line interface used to work with your computer

- shell = A program running on terminal

Bash = A popular shell on windows os
z-shell = Another shell (default of Mac)

* git bash installation for windows

→ download .exe from git-scm.com/download/win

→ check on on the desktop in Select components window
→ select overside the default branch name for new repositories and write main in branch selection for git window
→ select use Git and optional unix tools from the command Prompt
→ next, next ..... install

* Basics

ls = list files from directory
clear = clear screen
pwd = print working directory
cd = change directory
cd .. = back button
↑ space
/ = root directory
~ = home directory

e.g. Computer name = HP@DESKTOP-2DHQ293
we will use HP@DP

HP@DP MINGW64 ~ } Git Bash screen
$
↑ we type here
output will be here

→ HP@DP MINGW64 ~
$ git --version
git version 2.42.0.windows.2

→ ls
'3D Objects'/
'Activator 12345.rar'
AppData/
'Application Data'@
Contacts/
Cookies@
Desktop/
⋮

→ pwd
/c/Users/HP

* Navigation

→ cd Downloads
HP@DP MINGW64 ~/Downloads
$

→ pwd
/c/Users/HP/Downloads

→ ls
62565.json
Git-2.42.0-64-bit.exe*
desktop.ini
projects/

→ cd ..
HP@DP MINGW64 ~
$

→ cd ..
HP@DP MINGW64 /c/Users
$

→ cd HP/Downloads
HP@DP MINGW64 ~/Downloads
$

→ cd ../..
HP@DP MINGW64 /c/Users
$

→ cd /
HP@DP MINGW64 /
$

→ cd ~
HP@DP MINGW64 ~
$

* Path

Absolute path = fixed path
Relative Path = Depends on current directory
(above example of cd)

→ cd /c/Users/HP/Downloads/projects
HP@DP MINGW64 ~/Downloads/projects
$

* Making Directories

→ mkdir = make directory (folder)
→ same folder will not be made.
can use absolute or relative path

→ mkdir /c/Users/HP/newfolder

* Flags

→ Flags are characters that we pass with commands to modify their behaviour

→ man ls = In output Terminal will show manual (type like documentation) of ls command. Means how ls command works
- In addition it show how its flags associated with ls command will work
- To quit manual press keyboard key Q.

→ Windows git bash doesn't have man command but it uses --help flag.

e.g ls --help
pwd --help
cd --help

→ some ls flags : -s or --size = displays size of files ~~or directo~~ in KB

ls -s or ls --size

: -l = displays ; size, date, user, ...
: -a = displays hidden files, too.
: -la = combines -l & -a both ~~files~~ flags

Touch command

→ touch filename.extension : creates a new file

→ touch index.html
ls
3D Objects/
'Activator 12345.sar'
"AppData"/
⋮
index.html
⋮

→ touch abc (can also creates file without extension)

→ touch = change file access and modification times.
- If any file does not exist, it is created with default permission.

Deleting Files & Folders

→ rm : removes files
→ rmdir : removes empty folders
→ rm -rf : removes any folders

→ This is permanent removal

→ rm abc

file names

→ rm index.html style.css app.js

→ rm -rf new

↑ directory name

-r : recursive

-f : forced