Notes 2: The Linux FS

- Presentation
- article

The File System

The way files are stored and organized.

Basic Concepts

- **The root directory:** The first directory in the filesystem that contains the entire filesystem represented by "/".
- **Current working directory:** Also known as the present working directory. It is the directory where you are currently working in. You are always working from a directory.
- **Parent Directory:** a directory containing one or more directories and files.
- **Child directory:** a better name for this is a subdirectory or subfolder. This is a directory inside another directory. See image for visual reference.
- **YOUR HOME DIRECTORY:** This is your user's personal directory where all your files are located. Every user has it's own home directory just like in a apartment complex they all residents have their own apartment while sharing the common areas. You have total ownership of your home directory but outside of the home directory only the root user can make changes. An example absolute path, assuming that user name is maria53, would be `/home/maria53`
- **The home Directory:** This is the parent directory of all the home directories. This is where all the users' home directory are. The absolute path of this directory is `/home`. Noticed that it starts at the root.
- **The root user:** This is the administrator account of your system. To perform administrative tasks or to manage directories and files outside of your home directory, you must obtain root privileges using the `sudo` command at the beginning of the command.
- **Path name:** Also known as file path. This is the location of a given file in your computer. A path name can be absolute path or relative path.
- **Absolute Path:** The location of a file starting at the root of the file system. For example, `/home/maria53/Downloads/list.txt` is the absolute path of the file `list.txt`. The advantage of absolute paths is that they can be used at any point of the file system regardless of your current directory. Any command that is given an absolute path will be able to find the file because it will start at the beginning of the filesystem. The disadvantage is that a command can be long to type if the file path is long.
- **Relative Path:** The location of a file starting from a child directory of the current working directory or from the current directory itself. The advantage of using relative path is that typing commands is faster. The disadvantage of relative paths is that they cannot work from anywhere in the filesystem. In order for a relative path to work, a file must be reachable from the current directory onwards. Another disadvantage of relative paths is that they require a better mental understanding of the linux filesystem in the sense that you must keep a mental image of the directory tree that you are working with. An example of a relative path would be `Downloads/list.txt` assuming that the current working directory is `/home/maria53`.

Bash Special Characters:

Special characters are function like commands that tell the shell to perform a specific action without having to type the complete command. These special characters make working on the command line more efficiently. Here is short list to keep in mind.

- **. (single period):** represents the current directory.
- **.. (2 consecutive periods):** represents the parent directory.
- **~ (tilde character):** expands the current users home directory. It is like a variable that the shell uses to store the absolute path of the user's home directory. This `~/Downloads` is the same as typing `/home/maria53/Downloads`
- **/ (one forward slash):** as mentioned earlier, this is the root directory and the shortest path in the system. This is the beginning of the directory tree. There is nothing before it and everything after it.
- **- (hyphen-minus):** is used to move to the previous current working directory.
- **# (hash or number sign):** This is used for single line comments in shell scripting.
- **! (single exclamation mark):** used for repeating a command from the history. For example !5 will repeat the 5th command in the command history. To view the entire command history type `history`.
- **!! (2 consecutive exclamation marks):** are used for repeating the previous command. For example, `!!` will repeat the previous command while, `sudo !!` will repeat the previous command but will add `sudo` at the beginning of the command. This is useful for times when we forget to type sudo when performing administrative tasks.

Bash Environment Variables

- **What is a variable?**
  - In programming, a variable is place to store data. A variable is like a box with a label. For example, if you a lot of pens in your desk and you place them in a box a label it pens, now the box store your pens. In programming a variable can be used to store temporary or permanent information that you will continuously reuse in your program. For example, `username='maria53'` the variable name now stores the value `maria`. When evener the programs need to access the maria's username, it can do it by referencing the variable `username`.
- **What is an environment variable?**
  - Environment variables store values of a user's environment and can be used in commands in the shell. These values can be unique to the user's environment which makes them ideal when writhing commands that you want to use regales of which user is using the computer. To see a list of your environment variables type `env`. To use the value stored in an environment variable you must prepend the variable name with a `$`. Here are some useful environment variables:
  - **$USER** = stores the current's user username
  - **$HOME** = stores the absolute path of current's user home directory
  - **$PWD** = stores the absolute path of the present working directory.
  - **$OLDPWD** = stores the absolute path of the previous current working directory

Commands to navigate the linux filesystem

To navigate the linux filesystem, you only need to master 3 commands: `cd, pwd` and `ls`. While there are other useful commands, those can be considered complimentary and you should learn them after you have mastered those 3.

PWD

- **Usage**
  - Displays the absolute path of the current working directory.
- **Formula**
  - `pwd`
- **Examples**
  - Print the absolute path of current working directory
    - `pwd`

CD

- **Usage**
  - Changes the current working directory. In other words, it moves you from one directory to another. By default, it will always send you to your home directory.
- **Formula**
  - `cd + destination absolute path or relative path`
- **Examples**
  - Go (change your current directory) to your home directory (there is more than 1 way of doing this):
    - `cd` (without any arguments, cd will take you home)
    - `cd ~` (using the ~ special character. as ~ will expand to the absolute path of the user's home directory)
    - `cd $HOME` (using the $HOME environment variable)
    - `cd /home/$USER/Downloads` (using $USER environment variable in the path)
  - Go to a specified directory with absolute path:
    - `cd /usr/share/themes`
  - Go to a specified directory with relative path assuming your current working directory is `/home`
    - `cd maria53/Downloads/`
  - Go to the previous working directory. This is useful when you are working with 2 directories located far in the directory tree
    - `cd -`
  - Go to the previous directory in the directory tree. One directory above.
    - `cd ../`
  - Go to 2 directories above the directory tree
    - `cd ../../`

LS

- **Usage**

    - ls is used for listing files and directories. By default it will list the current directory when no directory is specified. Listing means to see what is inside a directory.
- **Formula**
    - `ls + option + directory to list`
- **Examples**
    - See all the options of the ls command (extracted from the man page):
        - `ls --help`
    - List the current directory:
        - `ls`
    - List all the files including hidden files in current directory:
        - `ls -A`
    - List all the files inside a given directory:
        - `ls -A /usr/share/fonts/X11` (absolute path)
        - `ls -A Documents/` (relative path assuming that the `$PWD` is `$HOME`)
    - Long list a directory
        - `ls -lA ~/Pictures`
    - List a directory recursively
        - `ls -R Documents/`
    - Long list a directory only
        - `ls -ld Documents/`
    - List a directory sorted by last modified
        - `ls -t Documents/`
    - List a directory sorted by file size
        - `ls -S Documents/`
    - Long list a directory excluding group and owner information, with human readable file size and sorted in reverse order.
        - `ls -lhgGr Documents/`

Tree

- **Usage**
    - used to display a recursive directory listing (tree) of files.
- **Formula**
    - `tree + option + directory`
- **Examples**
    - Display a tree of the home directory
        - `tree ~` or `tree $HOME or tree /home/$USER`
    - Display a tree of a directory with file permissions
        - `tree -p ~/practicels`
    - Display a tree of a directory with the full path prefix, with the user and group owner, human readable file sizes, and the date of last modified.
        - `tree -pughD ~/practicels/`

EXA

- **Usage**

    - Used the same way as ls. To list files and directories. Exa is a modern replacement for ls written in Rust. Exa is not installed by default. You will need to install it using your linux distribution package manager.
- **Formula**
    - `exa + option + directory`
- **Examples**
    - list all files in a given directory in a single line
        - `exa -1 ~/practicels`
    - long list all the files in a given directory
        - `exa -l ~/practicels`
    - long list all the files in a given directory with headers
        - `exa -lh ~/practicels`
    - list a directory in a tree like format
        - `exa -T ~/practicels`