Pathnames in Bash

Get yourself acquainted with the necessary concepts related to paths and their types.

Before hopping on the Navigation commands, it is important to cover the terms and concepts necessary to understand those commands. This lesson, again, is optional if you are already familiar with the terms such as Relative and Absolute path and know what references are. If you have even a slightest of doubt about any of these concepts, then it is suggested that you give this lesson a thorough read. It will clear all your doubts and confusions and will help you understand it with simple examples. So let's get to it one by one:

What is a Path?

A path can be a location to any folder or subfolder in your file system. Any path starts with the slash symbol (/) and contains names of different sub-folders in the hierarchy. You have to add a slash after the name of every sub-folder that you mention in the path.

When you open your terminal, it by default points to your "home" folder which is usually referred as "current directory". If you want to run any file by its name only, you need to be the first present in that directory where that file is stored, by changing your current directory. Otherwise, you can only run that file by specifying the complete path.

Examples:

- /home
- /usr/bin/local/app

There are two types of paths:

- Absolute Path
- Relative Path

Absolute Path

The pathname in which you specify the complete path from root to where the file is stored is called *Absolute Path*. This pathname is a combination of folder names and slashes, along with the file name and its extension. Every object stored in your filesystem will have a unique pathname. Here are few examples to give a better understanding:

- /home/downloads/music.my_song.mp3
- /home/Documents/file.txt

> Note:
>
> The first `/` in a pathname means the root directory and each `/` that we add in the path desends to the sub-level in your file system.

Relative Path

A relative path starts from your present working directory (pwd) also called "current directory" instead of the root directory. So instead of specifying the whole path from root to the file, you just specify it from your current directory. This technique comes in handy when the files resides in one of the child directories of your current directory. For example, if there's a hierarchy of directories in your filesystem where **Download** is your current directory and you need to move to its child directory

> Note:
>
> In Unix, a `.` is used to represent the current directory at which you're standing and `..` means *parent directory*.