Command Line Guide

Description	Command (macOS, Windows w/ cmder)	Windows (w/o cmder)
Print the working directory path	`pwd`	`cd`
Change directory	`cd [destination]`	`cd [destination]`
Change directory to parent directory	`cd ..`	`cd ..`
List the contents of the current working directory	`ls`	`dir`
List the contents of a specific directory	`ls [destination]`	`dir [destination]`
Rename a file	`mv [currentFile] [desiredFile]`	`move [currentFile] [desiredFile]`
Move a file	`mv [source] [destination]`	`move [source] [destination]`
Copy a file	`cp [source] [destination]`	`copy [source] [destination]`
Refer to the parent directory	`..`	`..`
Refer to the current directory	`.`	`.`
Delete a file irrevocably	`rm [file]`	`del [file]`
Delete a folder and its contents irrevocably	`rm -rf [folder]`	`del [folder]`

Commands

Commands are used in the following fashion:
`[command] [flag(s)] [argument1] [argument2…] [argument3…]...`

Working Directory

When you use the command line, your commands apply to the current folder the terminal is "focused" on. That folder is called the working directory.

Paths

A path refers to a series of folders you progress through to get to a destination. Paths use slashes to separate folders. Depending on your operating system and terminal, these will be either forward slashes (/) or backslashes (\). On a Mac, the following path means that the file message.txt is inside a folder called project, which is inside another folder called Desktop, which is in a folder called Covalence, which is in a folder called Users, which is on the hard drive:

`/Users/Covalence/Desktop/project/message.txt`

Another way of thinking of this is "start at the root of the hard drive, go into the Users folder, then the Covalence folder, then the Desktop folder, then the project folder, then access message.txt". This is an example of an **absolute path**, because no matter what our working directory is, this path provides "absolute" instructions for getting to a file from the root of the hard drive. In other words, the path has a constant starting point.

Relative Paths

Most of the time, you will use relative paths. These are paths where the starting point is the current working directory. Relative paths can be used with commands to allow those commands to affect files/folders outside of the current working directory. Below are some examples of how you could delete message.txt from the example above. Each example is based in a different working directory so that you can see how the relative paths differ.

Working Directory: project `rm message.txt`

Since relative paths are steps to take to get to a file, relative to the current working directory, the relative path to get to message.txt is just `message.txt`, because we are already inside the `project` folder

Working Directory: Desktop `rm project/message.txt`

Since the starting point is the `Desktop` folder, we need to step into the `project` folder, and then access `message.txt`.

Working Directory: Covalence `rm Desktop/project/message.txt`

Working Directory: Users `rm Covalence/Desktop/project/message.txt`