Symbolic Links & Hard Links

Get yourself versed in symbolic links and their difference with hard links.

Before digging in further, let's discuss a few concepts which a programmer must be familiar with.

This lesson is entirely dedicate to links but to understand links, you must have an idea of what an inode means. An *inode* is basically a link that points to the actual content or data. A file in the file system actually points to this inode instead of pointing directly to data.

What are Symbolic Links?

Just like there are *shortcuts* in Windows and *alias* in Mac, symbolic links (aka Soft Link) is a file referring to another file. You can think of it as a pointer pointing to a another memory location just like in C++. If you delete the link, it will not affect the original file.

> Note:
>
> The symbolic link of a file will not work anymore, and hence become a "dangling" link if you move that file to another location.

In Linux, you can create a soft link using `ln` command.

```
ln -s <source_name> <link_name>
```

Difference between Symbolic Link & Hard Link

Contrary to Symbolic Link, there is Hard Link which is the exact copy of the orginal file. This means that the original file and its hard link point to the *inode* in the directory.

```
ln <source_name> <link_name>
```