**Absolute vs. Relative Paths**

**We've mostly been dealing with** [**relative filepaths**](https://www.redhat.com/sysadmin/linux-path-absolute-relative) **which are paths that take your current directory into account. For example, let's say we have the following directory structure in our filesystem:**

**vehicles**

**├── cars**

**│ ├── fords**

**│ │ ├── mustang.txt**

**│ │ └── focus.txt**

**When inside the top-level vehicles directory, the relative path to the mustang.txt file is:**

**cars/fords/mustang.txt**

**However, when we're inside the cars directory, the relative path to the mustang.txt file is just:**

**fords/mustang.txt**

**Or when inside the fords directory, just:**

**mustang.txt**

**Absolute Paths**

**An absolute path is a path that starts at the root of the filesystem. On** [**Unix-like systems**](https://en.wikipedia.org/wiki/Unix-like) **(macOS/Linux), the root is denoted by a forward slash /. So, if the vehicles directory is in the filesystem root, the absolute path to the mustang.txt file is**

**/vehicles/cars/fords/mustang.txt**

**So, when inside the fords directory, you can use either:**

**/vehicles/cars/fords/mustang.txt**

**or**

**mustang.txt**

**to refer to the same file.**

**Which Should I Use?**

**It depends.**

**Relative paths are easier to read and write, and as long as you're in the correct directory (or the directory you expect), they're easier to reason about.**

**Absolute paths are more explicit. They're useful when you're not sure what directory you're currently in. For example, maybe you're giving someone instructions on how to find a file on their computer. You can't be sure what directory they'll be in when they start following your instructions, so you'll need to use an absolute path.**