Help Topic: Command Line Survival Basics

Spring 2022

Michael Benjamin, mikerb@mit.edu
Department of Mechanical Engineering
MIT, Cambridge MA 02139

Command Line Quick Survival Basics

The Terminal, i.e., the command line environment, offers an incredibly powerful set of tools out of the box. These tools are readily extendible through the addition of countless open source additions, as well as through additions made by you, in this course and hopefully well afterwards. To get started, the below list represents the bare bones basics. This list represents my own perspective, but you can find similar material by just doing a web search on "command-line basics".

Here we use the terms *Terminal* and *command line* interchangeably. This reflects a Mac user's perspective since the *Terminal* application is the most common way to interact with the command line. If you're a GNU/Linux user, you're more likely to refer to the *Console* instead of the terminal.

Let's get started.

Viewing the Contents of Your Directory with `ls` Command

The command line offers just another way of navigating through the same file system you might otherwise navigate with GUIs such as Finder. Perhaps the most commonly used command is `ls`, which just displays the contents of present directory:

```
$ ls
mail  project_fango   public_html   report_a.pdf   report_b.pdf   test.txt
```

The `ls` command comes with a bunch of *command line options*. These are extra things you can type on the command line as arguments to the command being invoked. For example, in the above, six items are reported. It turns out that `mail`, `project_fango`, and `public_html` are directories, i.e., folders, containing other files or directories. This is hard to discern from the above output. But if you use a simple command line option, "-F", you'll be able to see this right away:

```
$ ls -F
mail/  project_fango/   public_html/   report_a.pdf   report_b.pdf   test.txt
```

Note all directories have the trailing "/" character. I also use `-G` to make the directories show up in color. This can be done by typing `ls -F -G`, or more simply, `ls -FG`. There are several useful options worth exploring when you get a chance. For example, in directories with many files, you can sort by file type or date. Or you can filter to only show files of certain types. To explore, just type `man ls` to see some options.

Note also that arguments provided on the command line are separated by spaces. This is why most command line users *loathe* file names with spaces. Note the names `report_a.pdf` above uses the underscore character instead.

Knowing what directory (folder) you are in with `pwd`

Using the command line, you are always "in" a particular directory. Remember, command line users tend to refer to a directory where other users might refer to a folder. To know what the "present working directory" is, simply use the "pwd" command with no arguments.

```
$ pwd
/Users/mary/lab_one
```

This is such a useful piece of information to have that many users customize their command line prompt (the dollar-sign at the left) to show the present working directory. As an exercise, search the web for "customize command line prompt" and see if you can set this up.

Changing Directories with the `cd` Command

Now that you know how to tell what directory you're, you might be wondering how to change directories, moving around the overall file structure. To do this we use the `cd` command. The most common argument is the full path name of the directory you want to go to, as in:

```
$ cd /Users/mary/lab_one
$ pwd
/Users/mary/lab_one
$ cd /Users/bob/projects/lab_three
$ pwd
/Users/bob/projects/lab_three
```

There are a few handy shortcuts that should be in your mental toolbox right away. The first is for changing to the directory right above your current directory. For example, suppose you are presently in the directory `/Users/bob/projects/lab_three`, and you wish to move up to the directory `/Users/bob/projects`. You could do it the hard way:

```
$ pwd
/Users/bob/projects/lab_three
$ cd /Users/bob/projects
$ pwd
/Users/bob/projects
```

Or you could do it the easy way, replacing the third line above with the simpler version below:

```
$ pwd
/Users/bob/projects/lab_three
$ cd ..
$ pwd
/Users/bob/projects
```

The `../` notation is an actual "directory" in every directory in your file system. This special shorthand name has been around forever. Another shorthand name is `./` which stands for the current directory. Because both directories begin with a dot, they are also known as *hidden* directories, and don't show up when you use the `ls` command. They can be seen if you use the `ls -a` option for showing "all" files and directories, including the hidden ones. Go ahead and try it now, you will see both `./` and `../` regardless of which directory you are in. Even the "root" directory, the top-most directory in the tree, contains the `../` directory. Trying to go up from the root directory will just land you back in the root directory however.

A few other handy shortcuts are below. To move up two directories at once:

```
$ cd ../../                    (go up two directories)
```

To move to straight to your home directory:

```
$ cd                           (go directly to your home directory)
$ pwd
$ /Users/bob
```

To move to a directory within your home directory:

```
$ cd ~/project                 (go to a subdirectory of your home directory)
$ pwd
$ /Users/bob/project
```

To move to the last directory you were in, prior to the last invocation of `cd`:

```
$ pwd
$ /Users/bob
$ cd /Users/mary/projects/one
$ pwd
$ /Users/mary/projects/one
$ cd -                         (go to the directory prior to the last cd)
$ pwd
$ /Users/bob
```