**Jupyter slideshow:** This notebook can be displayed as slides. To view it as a slideshow in your browser, type the following into the console:

> ipython nbconvert [this\_notebook.ipynb] --to slides --post serve

To toggle off the slideshow cell formatting, click the CellToolbar button, then View --> Cell Toolbar --> None.



Using the Command Line[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Using-the-Command-Line)

*Author: Kiefer Katovich and Dave Yerrington | DSI-SF*

Learning Objectives[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Learning-Objectives)

*After this lesson, you will be able to:*

* Create folders and files using the command line (mkdir, touch).
* Change directories and list directory content (cd, ls).
* Check the current working directory (pwd).

Student Pre-Work[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Student-Pre-Work)

*Before this lesson, you should already be able to:*

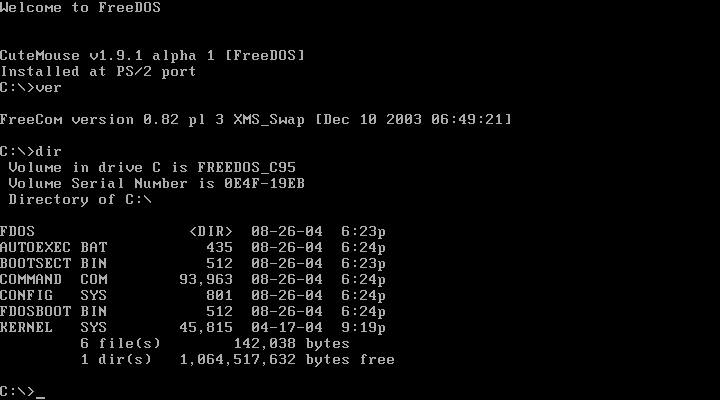
* Complete the [GA pre-work](http://generalassembly.github.io/prework/cl/#/).
* Open the terminal.
* Familiarize yourself with the [UNIX](http://mally.stanford.edu/~sr/computing/basic-unix.html) commands cd, pwd, mkdir, pwd, and touch. (Don't worry, we'll be going over these commands in class, just make sure you are able to describe what each of them means).

Table of Contents[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Table-of-Contents)

* [Introduction](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#introduction)
* [Command Line](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#command_line)
* [Paths](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#paths)
* [Editing and Examining Files](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#editing_files)
* [Finding Files](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#finding_files)
* [Independent Practice](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#independent_practice)
* [Intro to IDEs](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#ide)
* [Conclusion](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#conclusion)

What is a GUI (pronounced "gooey")?[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#What-is-a-GUI-(pronounced-"gooey")?)

There was a time when computers didn't come with a Graphical User Interface (GUI). Instead, everyone interacted with the computer using text commands in what we call a Command Line Interface (CLI).

*Image from Wikimedia*

Today, the command line still exists, even though you may have never seen it as a casual computer user. However, knowing basic CLI commands is a useful skill.

This CLI is the gateway to your operating system and can perform many of the same actions without a graphical interface. It is also facilitated by what we call a "shell."

Why Command Line?[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Why-Command-Line?)

Everything you can do in a windowed environment, you can do in the terminal — and faster. When you use the mouse, click on buttons, confirm operations, or wait for windows to load, you waste time! Terminal commands can assist you with:

* Running processes
* Finding files
* Substring match of file contents
* Assessing performance
* Remote operations
* Web browsing
* Installing packages
* Managing your development environment

What is a Shell?[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#What-is-a-Shell?)

A shell is simply a type of command line program that contains a simple, text-based user interface, enabling us to access all of an operating system's services. It is, put simply, a program that accepts text as an input and translates that text into the appropriate functions you want your computer to run.

*Taken from Just for Fun:* [*Type like a hacker*](http://hackertyper.com/)

Here are a few Windows equivalents: [Cygwin](https://www.cygwin.com/) [msis git](https://msysgit.github.io/)

[Windows shell equivalents](http://stackoverflow.com/questions/28487128/what-program-in-windows-is-equivalent-to-oss-terminal)

Running applications, checking settings such as free hard drive space, and even web browsing can all happen from the shell.

Forget Finder, Get Fast at Using Your Laptop[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Forget-Finder,-Get-Fast-at-Using-Your-Laptop)

Opening & Closing Terminal[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Opening-&-Closing-Terminal)

First, we need to launch the command prompt. We do this by using Spotlight:

* ⌘ (Command) + Space
* "*Te*rminal"
* Enter

Notice that you can actually hit enter as soon as the field autocompletes. Get used to taking shortcuts — don't type the whole word out if you don't have to, and avoid using your mouse if you can open or use an app with just keyboard shortcuts. It may seem hard right now, but, when you get used to it, it will save you hours of time cumulatively.

Paths[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Paths)

Every file or folder in a file system can be read, written, and deleted by referencing its position inside that system. When we talk about the position of a file or a folder in a file system, we refer to its "path." There are a couple of different kinds of paths we can use to refer to a file — **absolute paths** and **relative paths**.

**Directory** is an important term that's used interchangeably with *folder*. Although they are not exactly the same, when we say "navigate to your project directory," think of it as "navigate to your project folder." Here's a little more information:

*Strictly speaking, there is a difference between a directory, which is a file system concept, and the graphical user interface metaphor used to represent it (a folder). If you are referring to a container of documents, then the term "folder" is more appropriate. The term "directory" refers to the way a structured list of document files and folders is stored on the computer. It is comparable to a telephone directory that contains lists of names, numbers, and addresses but does not contain the actual documents themselves.*

*Taken from* [*Close-To-Open Cache Consistency in the Linux NFS Client*](http://www.citi.umich.edu/projects/nfs-perf/results/cel/dnlc.html)

What is an absolute path?[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#What-is-an-absolute-path?)

All files can created, updated, or deleted using the command line interface. We do this by referencing **paths**; either relative or absolute.

An absolute path is defined as the specific location of a file or folder as accessed from the root directory, typically shown as /. The root directory is the starting point from which all other folders are defined and is not usually the same as your **home** directory, which is normally found at /Users/[Your Username].

Working with UNIX commands and file paths[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Working-with-UNIX-commands-and-file-paths)

Typing **cd** — a command for "change directory" — with no parameters takes us to our home directory.

cd

If we type in pwd — a command for "print working directory" — from that folder, we can see where we are in relation to the root directory. The pwd command will always give you the absolute path of your current location.

Here's an example of absolute path:

open /Users/Lucy/desktop/a/b/c/file.txt

Notice that this path starts from / directory, which is a root directory for every Linux/Unix machine.

What is a relative path?[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#What-is-a-relative-path?)

A relative path is a reference to a file or folder **relative** to your current position or the present working directory (pwd). If we are in the folder /a/b/ and we want to open the file that has the absolute path /a/b/c/file.txt, we can simply type:

open c/file.txt

or

open ./c/file.txt

We can also use the absolute path at any time by adding a slash to the beginning of the relative path. The absolute path is the same for a file or a folder, regardless of the current working directory, but relative paths differ based on directory. Directory structures are laid out like so: directory/subdirectory/subsubdirectory.

**Check:** What is the difference between an absolute path and a relative path?

Navigating using the command prompt[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Navigating-using-the-command-prompt)

* Changing directories
* Listing files
* Creating directories and files
* Removing files

The tilde ~ character is an alias for your home directory. Use it to quickly return home.

cd ~

Or, even more simply, you can type:

cd

The tilde ~ character is useful for shortening paths that would otherwise be absolute. For example, to navigate to your Desktop, you can type:

cd ~/Desktop

The ls command lists files and directories in the current folder.

ls

It can also be used to list files located in any directory. For example, to list your applications, you can type:

ls /Applications

To make a new directory, type:

mkdir folder

To create a new file, type:

touch file1

To remove a file, type:

rm file1

General format for commands[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#General-format-for-commands)

<command> -<options> <arguments>

* <command> is the action we want the computer to take
* <options> (or "flags") modify the behavior of the command
* <arguments> are the things we want the command to act on

Using wildcards in the command prompt[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Using-wildcards-in-the-command-prompt)

The wildcard symbol (\*) is useful for using commands to operate on multiple files. To provide an example, first create a folder on your Desktop and add some files.

mkdir ~/Desktop/example\_folder

cd ~/Desktop/example\_folder

touch cat.txt

touch dog.txt

touch bird.txt

touch fish.txt

You can use the wildcard \* to then operate on subsets of files. List any file with "i" in the file name, for example:

ls \*i\*

Or, remove any file with "d":

rm \*d\*

ls

Hidden directories can also be found[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Hidden-directories-can-also-be-found)

There are hidden directories all over your file system — mainly to save you from youself. Using the parameters -lha to ls, we can find these directories.

ls -lha

-l:    One entry per line.

-h:    When used with the -l option, use unit suffixes: Byte, Kilobyte, Megabyte, etc.

-a:    Include directory entries whose names begin with a dot (.).

Editing and Examining Files[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Editing-and-Examining-Files)

At times it's helpful to edit files in a pinch. We can use the terminal editor nano to accomplish this.

Editing files from the terminal with nano can be accomplished with the following syntax:

nano [filename]

These hotkeys are available:

* **ctrl-w**: Search within file.
* **ctrl-o**: Save file as [filename].
* **ctrl-x**: Exit editor.

*The bottom of the editor contains the most common operations.*

Echo file content to the terminal[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Echo-file-content-to-the-terminal)

Sometimes it's nice to view the contents of files as text. There are a variety of ways to do this. The commands cat, head, and tail will allow us to view the entire or partial contents of a target file.

cat /etc/passwd

*Traditionally, the /etc/passwd file is used to keep track of every registered user with access to a system.*

**Only the first few lines of a file**

This command is useful when looking at files that might be too large to open in a traditional editor such as Sublime or Atom.

head /etc/passwd

**Only the last few lines of a file**

tail /etc/passwd

*You can also pass the paramter -n to head and tail to control the amount of output that is displayed*

Searching inside files: grep[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Searching-inside-files:-grep)

The wonderful grep command will search within files and traverse within subdirectories.

**Find all files with the word "the" inside.**

grep -r "the" \*

*Omitting -r will cause grep to only look within the current subdirectory.* *Using -i will make grep ignore the casing of characters, but at the expense of efficiency.*

Finding files[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Finding-files)

By far, the most useful operation from the terminal is finding files. locate finds files all over your file system. The find command will find files relative to the current working directory but needs to be used in conjunction with a pipe operation.

Find all notebook files within subdirectories of the current working directory[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Find-all-notebook-files-within-subdirectories-of-the-current-working-directory)

find . | grep ipynb

Find specific file(s) within the entire system[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Find-specific-file(s)-within-the-entire-system)

locate nanorc

Find specific file(s) with a substring match[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Find-specific-file(s)-with-a-substring-match)

locate log

Misc trick: Counting the number of lines in a file[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Misc-trick:--Counting-the-number-of-lines-in-a-file)

At times, you may not want to load an entire file in memory. Using a combination of head, tail, cat, and a new command — wc — you can quickly assess the various size characteristics of any file.

Find the number of lines in a file[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Find-the-number-of-lines-in-a-file)

cat /etc/passwd | wc -l

Find the number of words in a file[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Find-the-number-of-words-in-a-file)

cat /etc/test.txt | wc -w

You probably have questions about "piping"...[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#You-probably-have-questions-about-"piping"...)

Here's some optional (but highly recommended) reading about pipe and I/O redirection on the command line:

* [I/O redirection](http://linuxcommand.org/lts0060.php)
* [Good examples of piping commands together](http://unix.stackexchange.com/questions/30759/whats-a-good-example-of-piping-commands-together)

Independent Practice[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Independent-Practice)

**Try out the mkdir, touch, cd, pwd, and ls commands on your own. If you want, try out using the wildcard command as well.**

Intro to IDEs[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Intro-to-IDEs)

In addition to being able to write commands in the Unix OS(mac) or Windows OS(PC) command langauge in the terminal, we can also execute commands in a variaty of languages like python, Java and Git in terminal/command line as well.

In your terminal you can enter into a python shell by simply typing python.

Within the python shell we can execute python commands

>>> *# assigning a variable*

>>> x = 'hello world'

>>> *# printing a variables contents*

>>> **print** x

hello world

Writing and trouble shooting alot of code in the terminal can be tedious as it is hard to write several line scripts. Almost all developers don't actually write their scripts in the command line, instead they use IDEs, Integrated Developer Environments to build and test their code.

Common IDEs for Data Science[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Common-IDEs-for-Data-Science)

The Anaconda package manager we installed earlier comes with two useful, python based IDEs, Spyder and Jupyter.

**Jupyter Notebooks**

Has cell based execution which allows you run run all the code in an entire cell simultaneously. Jupyter books also have markdown and slide show integration allowing for them to make great blog or instruction resources. *All of the lessons in this class have been and will most likely be written in a Jupyter Notebook*.

Jupyter Notebooks open in your default browser and can be opened from the Anaconda Navigator or from the command line by executing jupyter notebook.

* [28 Jupyter Notebook tips, tricks and shortcuts](https://www.dataquest.io/blog/jupyter-notebook-tips-tricks-shortcuts/)
* [Markdown Cheatsheet](https://github.com/adam-p/markdown-here/wiki/Markdown-Here-Cheatsheet)

**Spyder Notebooks**

Has a selection based execution which allows you to run all the code that you have *selected* simultaneously. Spyder notebooks are very similar to R Studio and have several customizable windows for displaying output, variables and computer usage.

Syder notebooks open in their own window and can be opened from the Anaconda Navigator or from the command line by executing spyder.

As far as notebooks go, one is not better than the other and it comes down to personal preferences as far as which one you use.

Conclusion[¶](http://localhost:8888/nbconvert/html/Desktop/SCHOOL_WORK/Week%201/your-development-environment-master/extra-materials/Command-Line/command-line.ipynb?download=false#Conclusion)

Today, we learned about the Command Line and common commands we can run through it. We also learned about file structures, absolute paths and relative paths.

Additionally we reviewed running python in the command line as well has more appropriate places to build codes (IDEs).

In time, you might find that simple operations are actually faster to perform from the command line.