Windows CMD Walkthrough

September 2018

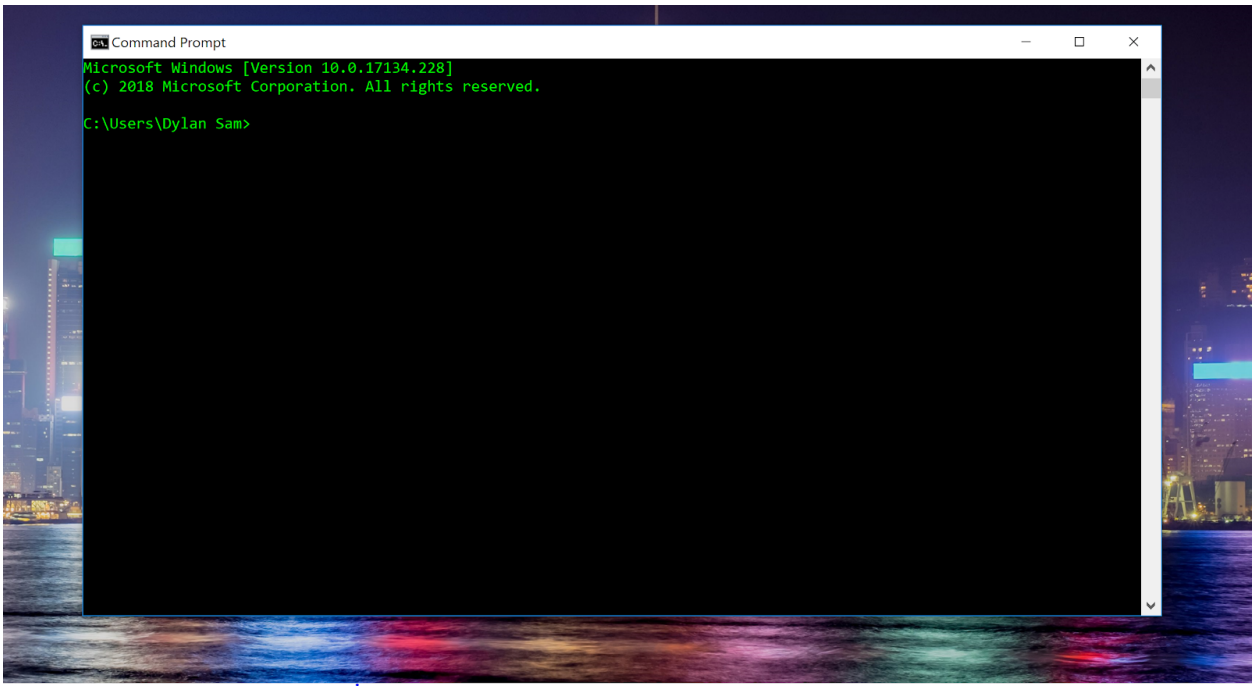
Contents

For windows, the command-line application is called a command prompt or 'cmd'. In this document, you will learn about files, directories, and many commands to help navigate them.

1 cmd

On the Mac, the command-line application is simply called Terminal. The name is a reference to computer terminals which were how many people, including university students, interacted with large computers before personal computers became commonplace.

If you type cmd in your Windows Search Box, you can open up something that looks like this:



1.1 Directories

While using the command line, youre always operating within a particular directory on your computer. This is called your current working directory. Its your current location, so to speak. You can navigate and operate within any directory. (In fact, just like the old Terminals referenced above, the functionality to fully control your system and do anything via the command line still exists.)

In cmd, type cd and press enter. This will print out your current working directory. cd stands for "change directory". When opening Terminal for the first time, you will always start in the same place, your home directory, which in my case is at C:\Users\Dylan Sam. My username is Dylan Sam, so you will see your own username at that place instead. From the command line, places where you store files are called directories. On Windows, C:\ or \both represent the root directory (the apex of the hierarchy of files that reside on your computer).

1.2 Listing Contents with dir

To view the files within your current directory you can type the command *dir*. This command stands for "directory". It will list the files in the directory. You can also tell dir to list the contents of a different directory. To list the contents of your Documents directory for example, type dir Documents. Here, we are giving the dir command a path. A path describes the location of a file or directory on our computer. There are two kinds of paths: relative paths and absolute paths. Our reference to Documents here is known as a relative path because we are specifying the path to this directory relative to our current working directory, C:\Users\Dylan Sam. You can also specify absolute paths by starting the full path with C:\.

Type dir "C:\Dylan Sam\Documents" but replace Dylan Sam with your username. You will see that the same files are listed.

This also shows how we specify nested directories, using the \to separate them. When specifying paths to a command in Terminal, we must be careful about directory names with spaces. In general, the Terminal will interpret any text following a space as another argument of input. For example, try typing dir Documents Pictures. It should print the contents of both the Documents and Pictures directory. If for some reason, you had a directory that was named "Documents Pictures", you would need to put that directory name in quotes dir "Documents Pictures".

1.3 Making Directories with mkdir

For the rest of the semester, we'll need a place to save our Python files. Create a new directory by typing mkdir "cs3_workspace". Here, mkdir stands for "make directory. We asked in our directory names, we replaced it with an underscore character ('_'), which is the command to create a directory called "cs3_workspace. To avoid the difficulty of spaces common practice.

1.4 Changing Directories with cd

Type the command *di* cs3_workspace. You should notice that the directory is empty. Now type cd cs3_workspace. Here cd stands for "change directory", and it changes the current working directory to wherever you specified as an argument. Type cd and it should now be C:\Dylan Sam\cs3_workspace, but with your own username.

We discussed how typing a directory name without a leading / or C: is a relative path.

But what if you want to specify a relative path to the directory one level up? Type cd .. and you should be brought to the directory that resides one level up. Type cd ..\.. and you will be brought to the directory two levels up, which should now be your root directory. Take a look at the files here. You will notice familiar directories like your "Applications" directory, and the "Users" directory that was listed when we typed pwd above. Type cd and you will be brought back to your home directory from anywhere, which is quite handy.

1.5 Running the Python Interpretor

Commands in the Terminal are just specially treated programs. You don't ever need to specify the path to these programs. ls , mkdir , and cd are examples of these programs. After we installed Anaconda, python is now included as a command we can use! Type python and you will see a small amount of output and now each line will start with >>>. This tells you that the Python interpreter will now understand each command as Python instead of the Terminal commands we were using before.

If you were to type one of the commands we talked about above, you would be told that it was not defined. This is because Python doesn't recognize the same commands as the Terminal. The Python interpreter is like its own little temporary world or playground. Temporary because you cant save any of the code you write within it, so when you exit it (via typing quit()), it all goes away. However, its a semi-convenient way to test small pieces of Python code before writing it to files and running it

While in the Python interpreter, type print("Hello").

Just below the line, you should see the output "Hello", exactly as you typed it. Now type print(100) without the quotes. You will see again that print outputs the value exactly as you put it in the statement. Do this again with print(3.14159). In Python, boolean values are case-sensitive and typed as True and False. Try printing out True or False as boolean value.

Once you are finished playing around with the interpreter, type quit() to leave.