Hyperiondev

TASK

Command Line

Visit our website

Introduction

Welcome to The 'Command Line' Task!

You are already familiar with front-end Web Development using HTML, CSS and JavaScript. In this level, you will learn to make use of tools like React and Express to create front-end and full stack web applications. As a web developer, it is essential that you are able to use web development libraries and frameworks to be able to build web applications. However, to do this, it becomes important to familiarise yourself with the Command Line. The Command Line is a tool that you will use often as a web developer. You will use the Command Line for many subsequent tasks.

Get in touch

Connect for support

Remember that with our courses, you're not alone! You can contact your mentor to get support on any aspect of your course.

The best way to get help is to login to **www.hyperiondev.com/portal** to start a chat with your mentor. You can also schedule a call or get support via email.

Your mentor is happy to offer you support that is tailored to your individual career or education needs. Do not hesitate to ask a question or for additional support!

WHAT IS THE COMMAND LINE AND WHY DO YOU NEED IT?

The Command Line is a means of interacting with a computer program where the user issues commands to the program in the form of successive lines of text. With the Command Line, you can quickly issue instructions to your computer getting it to do precisely what you want it to do. The Command Line is rarely used by most end users since the advent of the Graphical User Interface (a more visual way of interacting with a computer using items such as windows, icons, menus etc.).

For full stack Web Development, you will find it helpful to use the Command Line when interacting with your files, especially those created using web development frameworks and libraries such as Django, Express or React. You also need to be familiar with the command line to work with version control systems like git. Hence, this task will allow you to acquaint yourself with some of the basics of the Command Line.

A note from our coding mentor
Valerie

Not sure what a web framework is? This blog post by Hyperiondev provides an overview of what a web framework is and answers some other frequently asked questions about web frameworks.

FINDING THE COMMAND LINE

In Windows, you can simply click the Start menu and type `cmd` in the search box to locate the Command Line. Alternatively, the Command Line should be one of the options under 'Programs' and you can simply click on the application to open it.

With Mac OS, open the Command Line by opening the terminal. This can be done by opening the Applications folder, navigating to Utilities and then launching Terminal. Alternatively, you can search for "terminal" to find the application to launch.

COMMON WINDOWS COMMANDS

All commands that you will use with the Command Line have three parts: the utility, the flags and the arguments. The utility will always appear first. The other two parts have different rules, depending on which command you are using; you may not have to use any flags or arguments at all. For example, the following frequently used commands can be utilised without flags or arguments:

cd	Displays the name of or changes the current directory.
date	Displays or sets the date.
del	Deletes one or more files.
dir	Displays a list of files and subdirectories in a directory.
exit	Quits the cmd.exe (Command Line) program.
help	Provides help for Windows commands.
mkdir	Creates a directory.
rem	Renames a file or files.
rmdir	Removes a directory.
shutdown	Allows a proper local or remote shutdown of a machine.
type	Displays the contents of a text file.
ver	Displays the Windows version.

COMMON MAC OS/UNIX COMMANDS

Notice some of the most commonly used terminal commands below.

pwd	Print working directory. Displays the directory you are currently in.
cd	Change directory to the path specified.
touch	Creates a new file.
rm	Removes a file or directory.
ls	Displays a list of files and subdirectories in a directory.

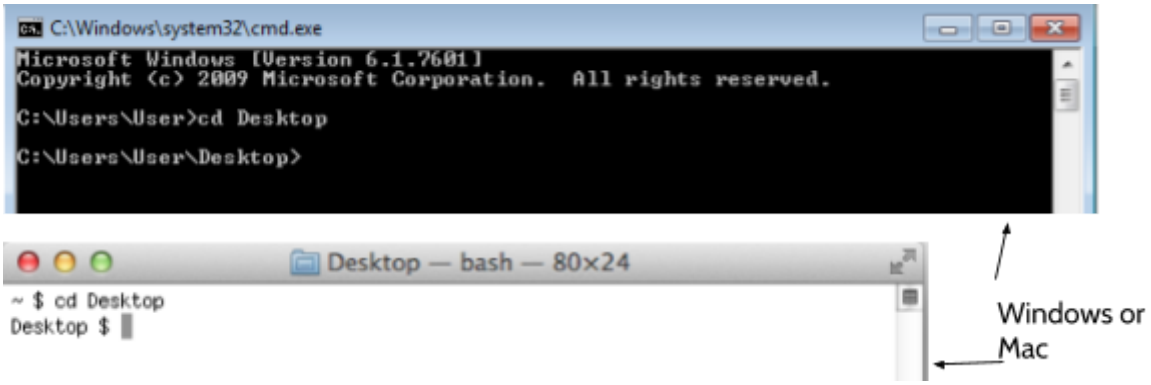
q	Quits the terminal.
mkdir	Creates a directory.
mv	Moves/renames a file.
man	Show the help manual for a command.
whatis	Provides a one-line description of what a command does.

As you can see, the Command Line has the built-in **`help`** (Windows) or **`man`** (Mac OS/Linux) command. This can be used to view all the commands that are executable. At this point, why not type the **`help/man`** command into the command line of your computer and hit Enter to find out more about all the commands? To get help on a specific command, you have to type **`help`** followed by the command in Windows (like so):

OR type **`man`** followed by the command in Mac OS/Linux.

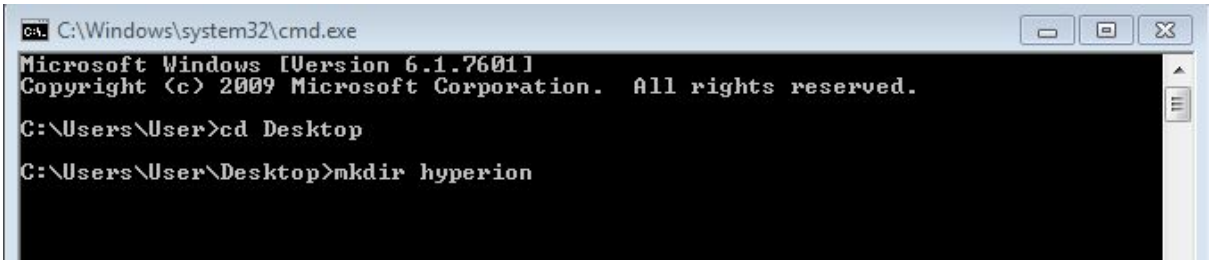
You could also type **`whatis`** followed by the command in Mac OS/Linux to get help. Compare the output you get with the **`whatis`** command with the output from the **`man`** command.

The command (in the images above) will give you the information about the **cd** command. As will be noted by the information provided by the Command Line, the **cd** command is used for navigation. It takes you from one directory to the next. For example, say you want to perform some command on a folder that is on your Desktop, you would have to type **cd** to change directory to your Desktop as shown in the images below.

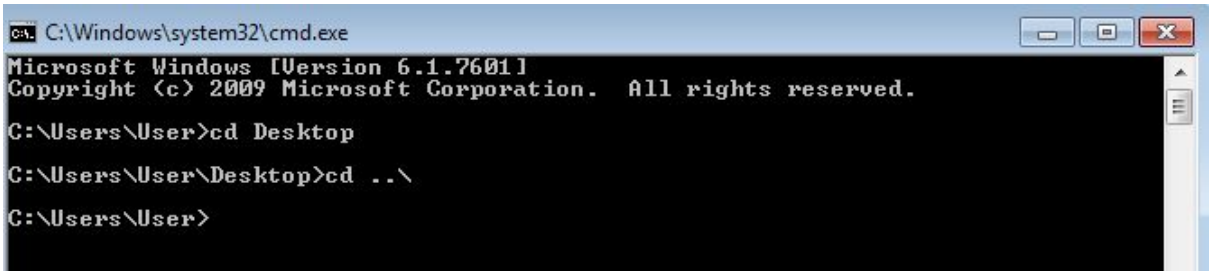


From here, we can now perform operations on the files or folders in our Desktop, since we have navigated into it. But, what if we have forgotten the name of the file or folder that we wanted to operate on? Well, you can simply use the **`dir`** (Windows) or **`ls`** (Mac OS/Linux) command to get a result of all the files or folders saved on the Desktop.

But, let's not alter any file or folder on the Desktop; instead let's create a new folder. Do you recall the command to use to make a new folder?

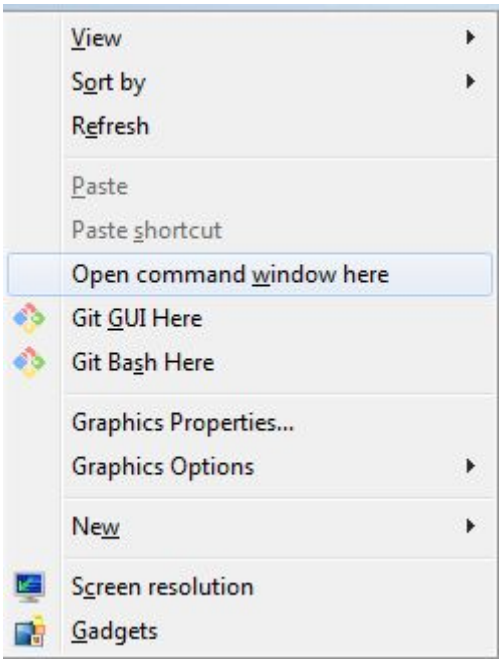


Notice that we have made a new folder on the Desktop called 'hyperion'. It's that simple! So, now that we have done what we wanted to do on our Desktop, how do we get back to where we were i.e. how do we navigate backwards?



To navigate two directories back, we would have to type, cd ..\..\. However, navigating back and forth may seem tedious to do. Wouldn't it be nice if we could

figure out a way in which we could open a Command Window in any directory with minimal effort? Fortunately, you can with Windows! Simply hold shift and right-click on a folder or empty space to open a Command Window in that directory.



SCRIPT FILES

As you advance in your skills as a web developer, you may at times find that there are certain commands that you use repeatedly. Instead of retyping these commands into the Command Line repeatedly, you can create a script file that contains these sets of commands and that can be executed as needed. Often such files will be executed periodically, e.g. daily, weekly, monthly etc. In Windows, we can create batch files and in Mac and Linux systems, we create shell scripts.

Batch files

A batch file is a kind of script file in DOS, OS/2 and Windows. Batch files are normally used by individuals who run the same commands frequently. Thus, instead of typing out the commands each time, the commands are simply placed in a batch file. To execute the commands contained within a batch file, you can simply double-click it.

The batch file consists of a series of commands to be executed by the Command Line, stored in a plain text file. To create a BAT File, you have to open a plain text editor (e.g. Notepad) and navigate to File > Save as, and in the “Save As” window,

input the name for your BAT File and then add a ".bat" extension.

Shell scripts

To create a shell script:

1. Open a text editor (e.g. gedit).
2. Add the following instruction: `#!/bin/bash` to the first line of the script file.
3. On the following lines enter the instructions that you would usually type into the terminal, one line per instruction.
4. Save the file. It is not a requirement but it is common practice to save your file with a .sh extension. To save the file properly you may need to specify that the file is a plain text file. Do this by selecting Format>Make plain text.
5. Make this file executable by typing the following into the command line: `chmod +x myscript.sh` where myscript.sh is the name of the script file.
6. To run the script type: `sh myscript.sh` where myscript.sh is the name of the script file.

Instructions

- The directory called “Examples” contains examples of a batch file (if you are using Windows) and a shell script called “MacExample.sh” (if you are using Mac OS). Please read through the comments in the example file that is relevant to you (based on the operating system you are running on your PC) before attempting this task.

- The additional reading (which is from the following source: http://abacus.gene.ucl.ac.uk/software/CommandLine.MACosx.pdf) is optional reading. However, we do recommend that you consult this brief guide for further information if you get stuck.

- Feel free at any point to refer back to the material if you get stuck. Remember that, if you require more assistance, our mentors are always more than willing to help you!

Compulsory Task 1

Follow these steps:

- Create a script file called ‘file_cd’. (Remember to save this file with a .bat extension if you are using Windows)
    - Inside ‘file_cd’, insert commands to create three new folders. Choose some cool names for your folders.
    - Next, insert commands to navigate inside **one of the folders you created** and create three new folders inside this folder. Also, insert commands to remove two of the directories you created.

- Create a batch file called ‘ifExample’. (Again remember to save this file with a .bat extension if you are using Windows) Inside it, add an if statement to make a new folder called ‘if_folder’ if one of the folders you created is named ‘new_folder’.

- Next, add an if-else statement that makes a new folder called ‘hyperion’ if an ‘if_folder’ exists, or else make a new folder called ‘react-projects’ if it does not.

  You would need to employ the following syntax for these conditional statements:

  If exist *filename command*
  If exist *filename (command)* else *(command)*

Once you have completed the task in line with the instructions above, click the button below to request your mentor to review your work and provide feedback. If you have any questions while attempting the task, leave your mentor a note on the comments.txt file in your Student Dropbox folder.

Completed the task(s)?

Ask your mentor review your work!

Review work

Rate us

Share your thoughts

Hyperion strives to provide internationally-excellent course content that helps you achieve your learning outcomes.

Think the content of this task, or this course as a whole, can be improved or think we’ve done a good job?

Click here to share your thoughts anonymously.