**Your First Command**

The **command line** is a text interface for your computer. It’s a program that takes in commands, which it passes on to the computer’s operating system to run.

From the command line, you can navigate through files and folders on your computer, just as you would with Finder on Mac OS or Windows Explorer on Windows. The difference is that the command line is fully text-based.

The advantage of using the command line is its power. You can run programs, write scripts to automate common tasks, and combine simple commands to handle difficult tasks – making it an important programming tool.

This course is for unix-based systems such as Linux and Mac OS X. An appendix of all commands taught in this course is available [here](https://www.codecademy.com/articles/command-line-commands).

**Instructions**

**1.**

To access the command line, we use a terminal emulator, often just called the *terminal*.

In the terminal, after the $, type:

ls

and press enter.

You should see three items print out below the command. Click Next to learn how this command works.

Hint

Be sure to type the letter l as in “lemon” and not the number 1.

After pressing enter, a new line will appear:

$ ls

2014 2015 hardware.txt

**NAVIGATION**

# ls

What’s going on here?

$ ls

2014 2015 hardware.txt

1. In the terminal, first you see $. This is called a shell prompt. It appears when the terminal is ready to accept a command.
2. When you type ls, the command line looks at the folder you are in, and then “lists” the files and folders inside it. The directories **2014**, **2015**, and the file **hardware.txt** are the contents of the current directory.

ls is an example of a command, a directive to the computer to perform a specific task.

**Instructions**

When using the command line, we refer to folders as directories. Files and directories on your computer are organized into a filesystem.

Click Next to find out how the filesystem works.

**Filesystem**

A filesystem organizes a computer’s files and directories into a tree structure:

1. The first directory in the filesystem is the *root directory*. It is the parent of all other directories and files in the filesystem.
2. Each parent directory can contain more child directories and files. Here **blog/** is the parent of **2014/**, **2015/**, and **hardware.txt**.
3. Each directory can contain more files and child directories. The parent-child relationship continues as long as directories and files are nested.

You’re probably already familiar with this tree structure - Mac Finder and Windows Explorer represent the filesystem as trees as well.

At any point, you can reference the filesystem for this lesson [here](https://s3.amazonaws.com/codecademy-content/courses/learn-command-line/img/LCL-fileTrees-01.png).

**Instructions**

**1.**

Let’s see how to navigate the filesystem from the command line. In the terminal, after the shell prompt, type:

pwd

and press enter.

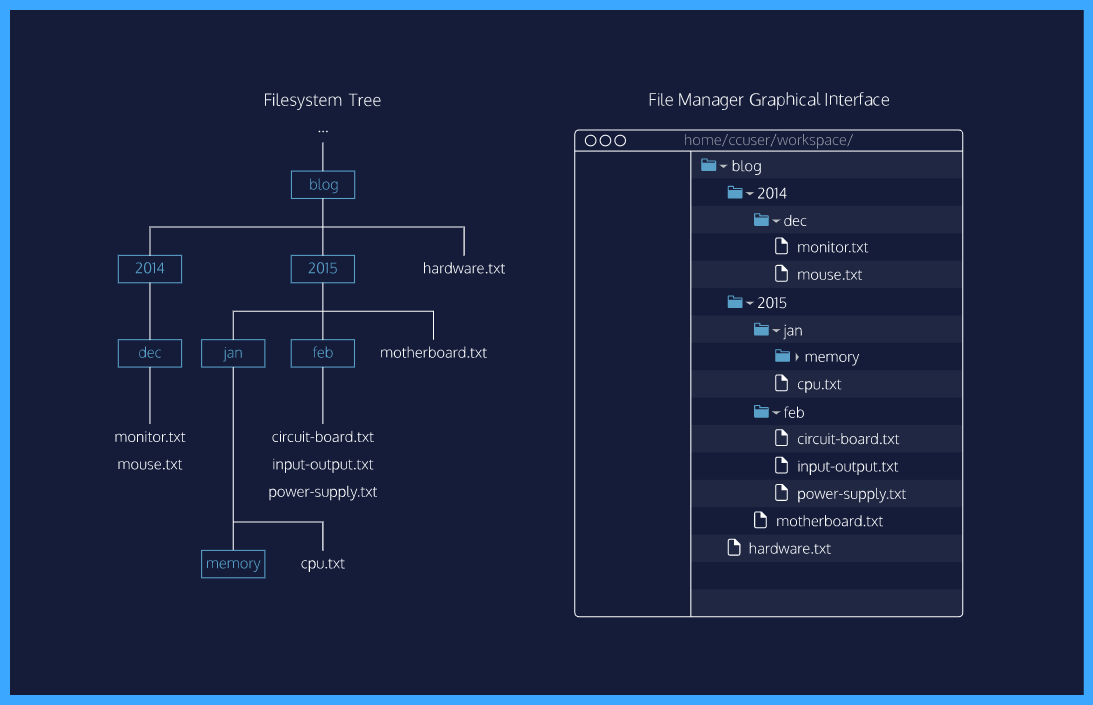
We’ll explain this in the next exercise.

Hint

After pressing enter, a new line will appear:

$ pwd

/home/ccuser/workspace/blog



# pwd

$ pwd

/home/ccuser/workspace/blog

pwd stands for “print working directory”. It outputs the name of the directory you are currently in, called the working directory.

Here the working directory is **blog/**. In Codecademy courses, your working directory is usually inside the **/home/ccuser/workspace/** directory.

Together with ls, the pwd command is useful to show where you are in the filesystem.

**Instructions**

**1.**

Let’s continue with more commands. In the terminal, print the working directory.

**2.**

List all files and directories in the working directory.

**3.**

Then type:

cd 2015

Again, print the new current working directory.

List all files and directories in the working directory.

**cd I**

$ cd 2015

1. cd stands for “change directory”. Just as you would click on a folder in Windows Explorer or Finder, cd switches you into the directory you specify. In other words, cd changes the working directory.
2. The directory we change into is 2015. When a file, directory or program is passed into a command, it is called an *argument*. Here the 2015 directory is an argument for the cd command.

The cd command takes a directory name as an argument, and switches into that directory.

Make sure you are in the directory **/home/ccuser/workspace/blog/2015** by using the command pwd.

**Instructions**

**1.**

Once you are in the directory **/home/ccuser/workspace/blog/2015**

Then type:

cd jan/memory/

Print the working directory to see the new location.

Hint

If you are in **/home/ccuser/workspace/blog/**, then enter the command cd 2015 before moving on to the checkpoints.

**2.**

Then type:

cd ..

Print the working directory again to see the new location.

# cd II

$ cd jan/memory

To navigate directly to a directory, use cd with the directory’s path as an argument. Here, cd jan/memory/ command navigates directly to the **jan/memory** directory.

$ cd ..

To move up one directory, use cd ... Here, cd .. navigates up from **jan/memory/** to **jan/**.

Make sure you are in the directory **/home/ccuser/workspace/blog/2015/jan** by using the command pwd.

**Instructions**

**1.**

Once you are in **/home/ccuser/workspace/blog/2015/jan**, change the directory to the **2015/feb/** directory using:

cd ../feb

List all files and directories in the working directory.

Hint

Type pwd into the terminal to see where you are.

If you are in **/home/ccuser/workspace/blog/**, then enter the command cd 2015 followed by: cd jan before moving on to the checkpoints.

**2.**

Type:

mkdir media

Again, list all files and directories in the working directory. You’ll see that there is now a new directory named **media/**.

# mkdir

$ mkdir media

The mkdir command stands for “make directory”. It takes in a directory name as an argument, and then creates a new directory in the current working directory.

Here we used mkdir to create a new directory named **media/** inside the **feb/** directory.

**Instructions**

**1.**

Navigate to the **2014/dec/** directory using cd.

List all files and directories in the working directory to see what is in there currently.

Hint

You can use pwd to see which directory you are currently in.

Make sure to navigate to that directory **in one step**, rather than going into the first directory, followed by the second.

**2.**

Then type:

touch keyboard.txt

Again, list all files and directories in the working directory. You’ll see that there is now a new file named **keyboard.txt**.

# touch

touch keyboard.txt

The touch command creates a new file inside the working directory. It takes in a filename as an argument, and then creates an empty file in the current working directory.

Here we used touch to create a new file named **keyboard.txt** inside the **2014/dec/** directory.

**Instructions**

The commands we’ve covered so far are commonly used to navigate the filesystem. There are more commands you can use to master the command line, and we’ll cover them in the next lessons.

Let’s summarize what we’ve done so far.

**Concept Review**

Want to quickly review some of the concepts you’ve been learning? Take a look at this material's [cheatsheet!](https://www.codecademy.com/learn/paths/web-development/tracks/learn-the-command-line/modules/learn-the-command-line-navigation/cheatsheet)

**Review**

Congratulations! You’ve learned five commands commonly used to navigate the filesystem from the command line. What can we generalize so far?

* The *command line* is a text interface for the computer’s operating system. To access the command line, we use the terminal.
* A *filesystem* organizes a computer’s files and directories into a tree structure. It starts with the *root directory*. Each parent directory can contain more child directories and files.
* From the command line, you can navigate through files and folders on your computer:
  + pwd outputs the name of the current working directory.
  + ls lists all files and directories in the working directory.
  + cd switches you into the directory you specify.
  + mkdir creates a new directory in the working directory.
  + touch creates a new file inside the working directory.

**Instructions**

Move on when you are ready!