Computer Science Activity—Navigating with the Command Line

BACKGROUND

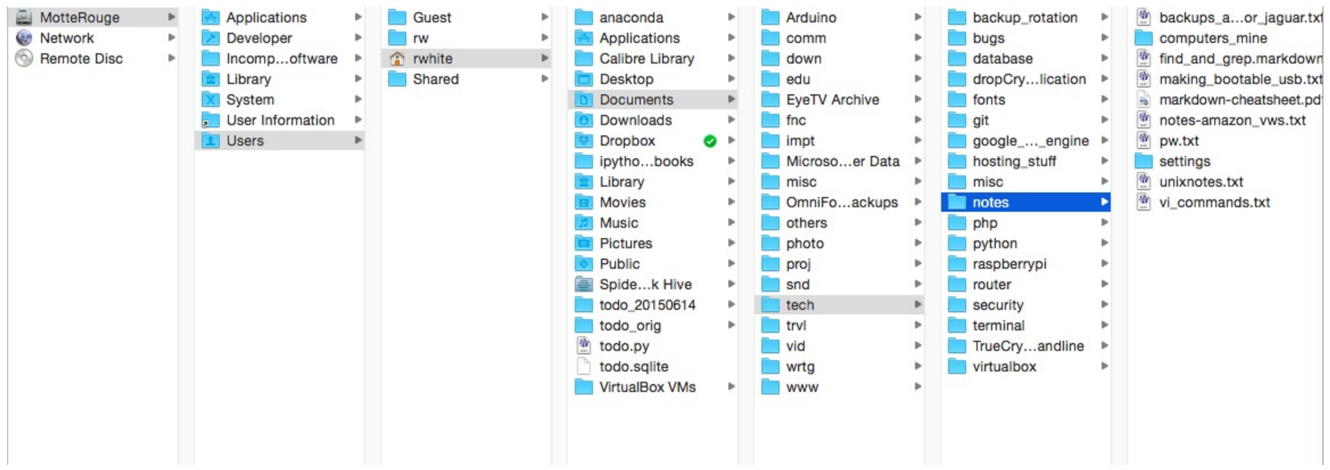
The files and directories ("folders") on a computer are organized in a hierarchical structure. Your operating system's graphical interface (Microsoft's *Windows*, Apples *OS X*, Ubuntu's *Unity*) allows you to intuitively interact with this hierarchy by clicking to select a file or folder, click-dragging to move a file or folder, and double-clicking to open a folder or to to open a file in its preferred application.

Before graphical interfaces were developed, interacting with the computer was done primarily via a *command-line interface* using a device or program call the *terminal*. This type of interface is still commonly used by computer scientists and programmers to interact with computers—once you know how to use it, it allows you to control your computer in ways that the graphical interface doesn't allow for.

To introduce you to the command line, we're going to learn a few commands to help us navigate your computer's file structure: `pwd`, `ls`, and `cd`.

VISUALIZING THE FILE STRUCTURE

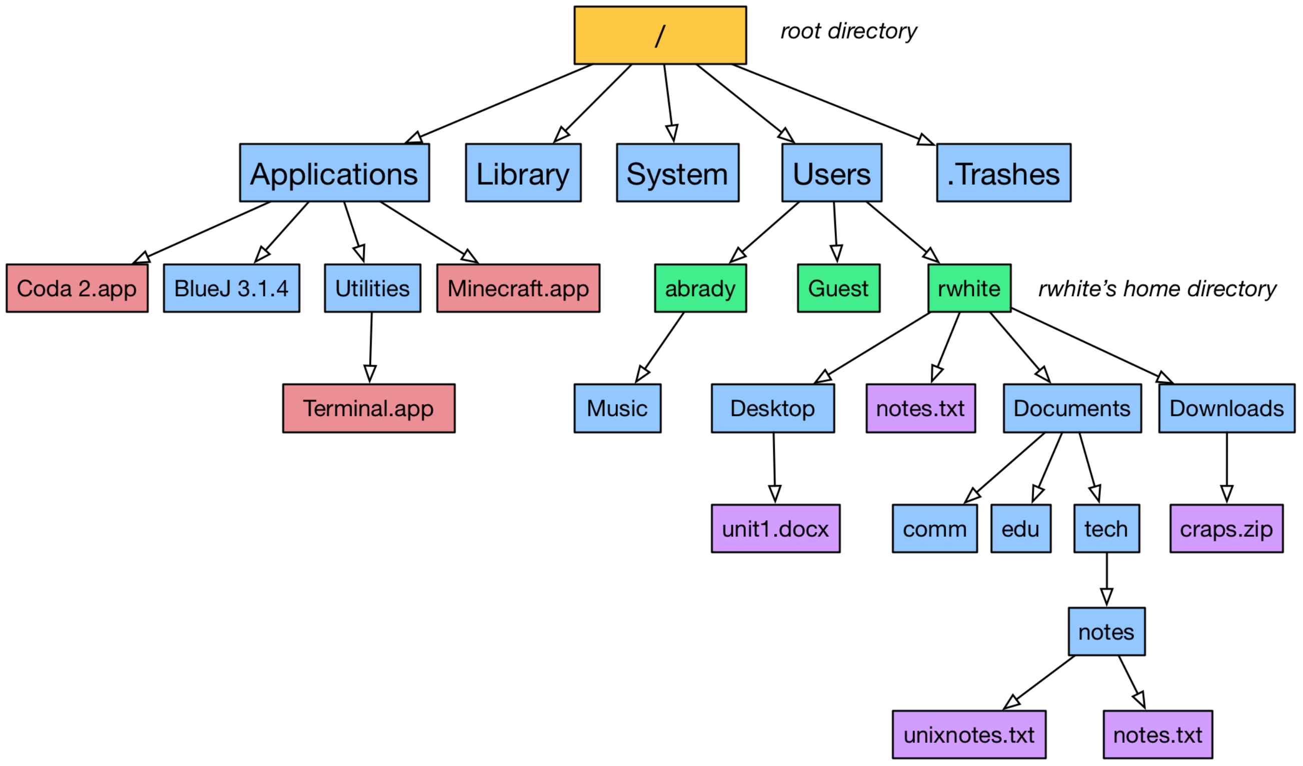
You've already seen graphical representations of the nested files and folders on your computer. Microsoft's *Windows*, Apples *OS X*, and any Linux window system allow you to view the contents of your computer's hard drive in a number of different ways. Here's one view of some nested files in Apple's OS X:



The computer is named `Motte_Rouge`, it contains seven directories including one called `Users`, and that folder in turn contains four directories including my home folder, `rwhite`, and so on.

On the next page is another way to visualize those same nested files and directories. Not all files and directories are shown in this diagram, but this is sufficient for learning the basics, and it's this file structure that we'll use in learning a few terminal commands.

It's important to keep these visualizations in mind when working in a command-line interface, because no such graphical representations are available to you there. You'll be issuing textual commands to navigate the file structure, and additional commands to work with files in that structure.



LAUNCHING THE TERMINAL

A *terminal emulator* is a program that allows us to use a *command-line interface* (CLI) with your computer. Both Apple OS X and Ubuntu/Linux operating systems come with a Terminal application pre-installed. OS X users will find the Terminal.app stored in the **Utilities** folder in **Applications**. Linux users will (probably) find a shortcut to launch the program in the **Menu/Accessories**. Windows users will need to install **Cygwin** on their computers to get the same kind of command line functionality. (Note that Windows' Command Prompt program offers some similar functionality, but for this exercise, Windows users will need Cygwin.)

When the Terminal first launches, some text will appear on the line, at the end of which will appear a `$`. The `$` prompt indicates that the Terminal is awaiting a command that launches a "utility."

```
$
```

Note that the Terminal typically doesn't allow you to use the mouse. You'll enter commands using the keyboard only.

ABSOLUTE and RELATIVE ADDRESSES

If I have two documents with the same name (`notes.txt`, say) stored on my computer, that's usually not a problem, as long as they are stored in different locations. There are two different ways to refer to the location of a directory or file.

Absolute address

The *absolute address* of a file or folder is given relative to the root directory `/`, which is the single directory that holds all others. Looking at the file structure above, the absolute address of one of the `notes.txt` files is `/Users/rwhite/notes.txt`

Relative address
When the Terminal is first launched, your location is in your *home directory*. For user **rwhite**, the absolute address of the home directory is `/Users/rwhite`, and you can confirm this by entering the command `pwd` in the Terminal, which will cause the current directory to be displayed. (In the examples given here, text in **bold** indicates commands entered by the user, while non-bold text is produced by the computer.)

```
$ pwd
/Users/rwhite
$
```

The *relative address* of a file or folder is given relative to where one is currently "located" in the file structure. The "period," or "dot" (`.`) is used to indicate the current directory, and a double-dot (`..`) indicates the directory *above* the current directory, called the *parent directory*.

The address for the `unit1.docx` file relative to the home directory of **rwhite** is:
`./Desktop/unit1.docx`
The address for the home directory of **abrady** relative to the home directory of **rwhite** is:
`../abrady`
The address of the Minecraft.app file relative to the home directory of **rwhite** is:
`../../Applications/Minecraft.app`

THE `ls` COMMAND
The `ls` command is used to list the contents of directories, specified by either *absolute* or *relative* addresses.

So if my current location is the home directory of **rwhite**, I can list the contents of the `Users` directory in one of two ways: an *absolute address* given relative to the root directory `/`, or a *relative address* given relative to the current directory (`rwhite` in this case).

```
$ ls /Users
Guest      abrady     rwhite
$ ls ../
Guest      abrady     rwhite
```

And I can list the contents of my `Downloads` folder in one of two ways:

```
$ ls /Users/rwhite/Downloads
craps.zip
$ ls ./Downloads
craps.zip
```

Note that a user's ability to list the contents of other folders may be limited by the permissions of those folders. One user, **rwhite** for example, typically doesn't have access to the contents of other user's directories.

```
$ ls ../abrady/Documents
ls: : Permission denied
```

To list the contents of the current directory, one would enter `ls .`   If you enter the command `ls` without the dot, the default directory to list is the current one:

```
$ ls .
Desktop     Documents   Downloads   notes.txt
$ ls
Desktop     Documents   Downloads   notes.txt
```

THE cd COMMAND
The `cd` command is used to change one's location to a different one in the file structure, as specified by either an *absolute* or *relative* address

So if my current location is the home directory of **rwhite**, I can move into other directories and issue commands at those locations as well.

```
$ pwd
/Users/rwhite
$ cd ./Documents/tech
$ pwd
/Users/rwhite/Documents/tech
$ ls .
notes
$ ls ./notes
unixnotes.txt     notes.txt
```

THE HOME DIRECTORY
In addition to the address references *root* (`/`), *current directory* (`.`), and *parent directory* (`..`), the tilde symbol (`~`) refers to the user's home directory. If I'm currently in the root directory:

```
$ pwd
/
$ ls /Users/rwhite
Desktop     Documents   Downloads   notes.txt
$ ls ~
Desktop     Documents   Downloads   notes.txt
$ cd ~
$ pwd
/Users/rwhite
```

FLAGS
Up to this point all of the commands that we have issued have include one or two components: the command itself, which launches a *utility* like `pwd`, `ls`, or `cd`, and in most cases an *argument*, information that the utility will be working on.

A third component to a command is often one or more *flags*, which can further specify how the utility should operate.

The `ls` utility, for example, can be issued with the `-l` flag, which causes the listing to be given in *long* form. Here's the *long* listing of the home directory.

```
$ ls -l ~
drwx------+ 22 rwhite  staff     748 Jun 24 20:11 Desktop
drwx------+ 23 rwhite  staff     782 May 24 08:56 Documents
drwx------+ 17 rwhite  staff     578 Jun 22 17:00 Downloads
-rwxr--r--   1 rwhite  staff   34851 Jun 15 07:24 notes.txt
```

Another flag for the `ls` utility is the `-a` flag, which causes *all* files in a directory to be displayed, including those that begin with a dot and are ordinarily hidden from view:

```
$ ls -a ~/Documents
.     .DS_Store        comm        tech
..    .localized  edu
```

Note that the current directory `.` and the parent directory `..` are included in the listing above.

Another useful flag for the `ls` utility is the `-R` flag, which not only lists the contents of the indicated directory, but will look into any directories contained there and list the contents of those as well. This `-R` flag will *recursively* list the contents of all enclosed directories. If I'm logged in as **rwhite**, then, I can list every file in my home folder:

```
$ ls -R ~
```

You can usually stack flags together, too. To list every single file, including hidden ones, in long form, in your home directory:

```
$ ls -alR ~
```

Depending on how many files you have in your home directory, listing these all out could take awhile!

GETTING MORE HELP

All of the utilities/commands available to you in the Terminal have a help file, or "manual page" that you can use to learn more about them. To view those pages in the Terminal, type `man <utility>` and the first screen of material on that utility will be presented.

```
$ man ls
```

To navigate through these pages, you use the following keys:

- `f` to move forward one page
- `b` to move backward one page
- `q` to quit the `man` documentation

There are other commands as well. How do you get help on the `man` command?

```
$ man man
```

QUESTIONS

A. On your own computer...

1. open up a Terminal and issue the command

   ```
   $ ls ~/Documents
   ```

   What are the names of the first five documents you see listed? (Feel free to skip omit any documents that are sensitive or that you don't wish to include here.)

2. Using the Terminal or your computer's file system, begin in your home directory and identify some of the directories and documents it contains, and some of the directories/documents contained within some of the directories. With this information, create a small map of some of your directories and documents similar to the map on page 2 of this handout.

B. Use the file structure given at the top of page 2 of this handout to answer the following questions. For some questions there may be several possible ways to answer correctly.

1. What is the absolute address of the `unixnotes.txt` file?

2. What is the address of the `craps.zip` file relative to the `edu` directory?

3. Immediately after launching the Terminal as **rwhite**, what command would you enter to list the contents of your home directory?

```
$
```

4. What command would you enter to list *all* the contents of the `edu` directory?

```
$
```

5. What command would you enter to move to the `Applications` directory?

```
$
```

6. From the `Applications` directory, what command would you enter to list the contents of that same directory?

```
$
```

7. From the `Applications` directory, what is the shortest command you could enter to list, in long form, the contents of `rwhite`'s `Downloads` directory?

```
$
```

8. What command would you use to immediately move to your home directory from anywhere else in the file structure?

```
$
```

9. What command would you use to list every single file on your entire computer?

```
$
```

10. What command would you use to get help on the `mkdir` utility?

```
$
```