



Introduction to the Command Line Interface

Jeffrey Leek
Johns Hopkins Bloomberg School of Public Health

What is the Command Line Interface?

Nearly every computer comes with a CLI

- Windows: Git Bash (See "Introduction to Git")
- Mac/Linux: Terminal

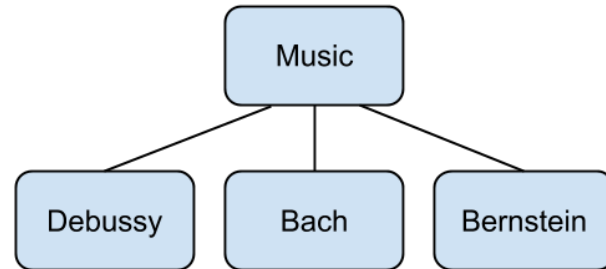
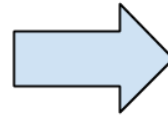
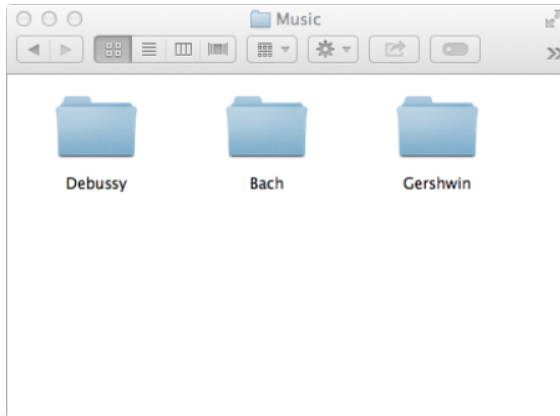
What can the CLI do?

The CLI can help you:

- Navigate folders
- Create files, folders, and programs
- Edit files, folders, and programs
- Run computer programs

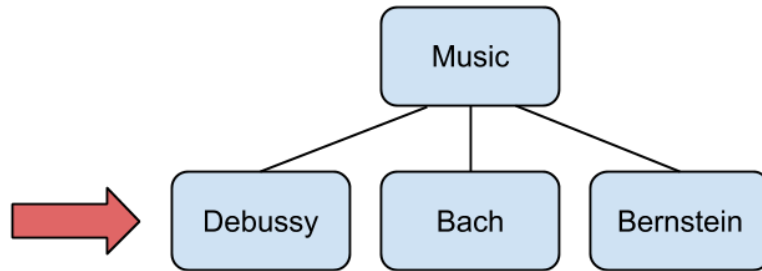
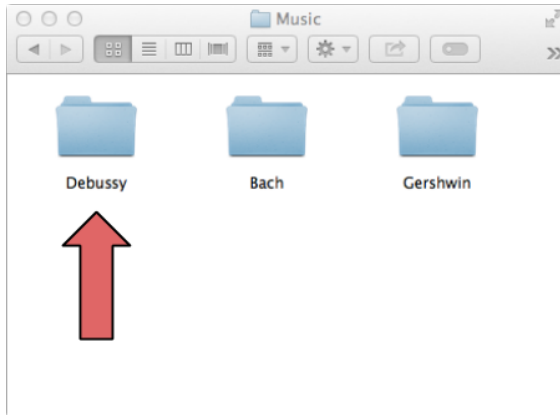
Basics of Directories

- "Directory" is just another name for folder
- Directories on your computer are organized like a tree
- Directories can be inside other directories
- We can navigate directories using the CLI



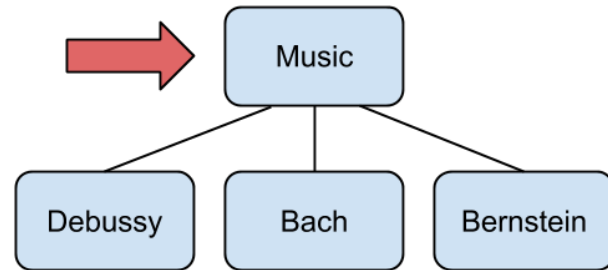
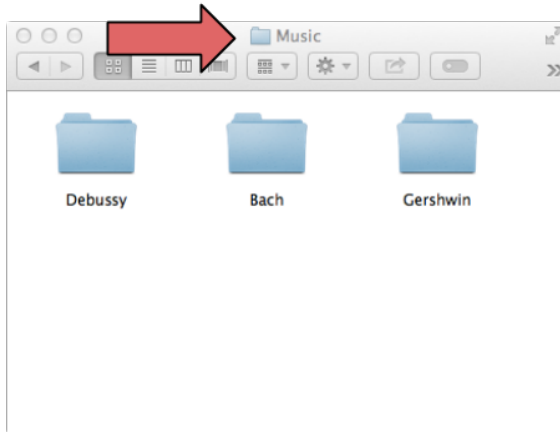
Basics of Directories

- My "Debussy" directory is contained inside of my "Music" directory



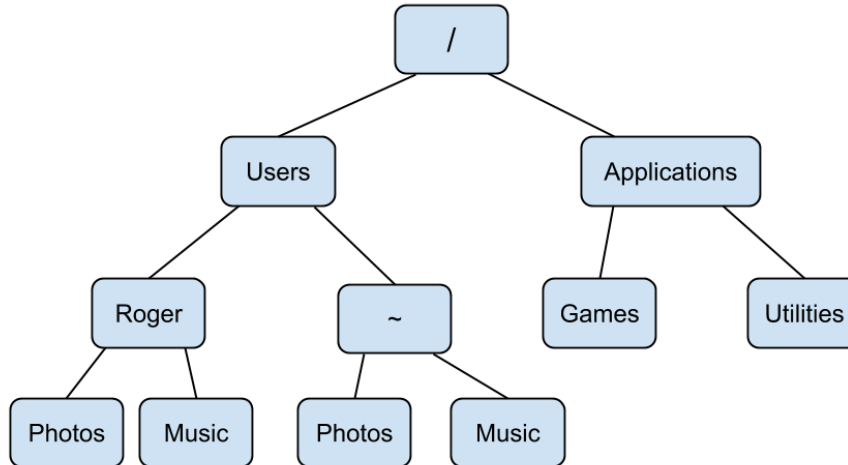
Basics of Directories

- One directory "up" from my Debussy directory is my Music directory



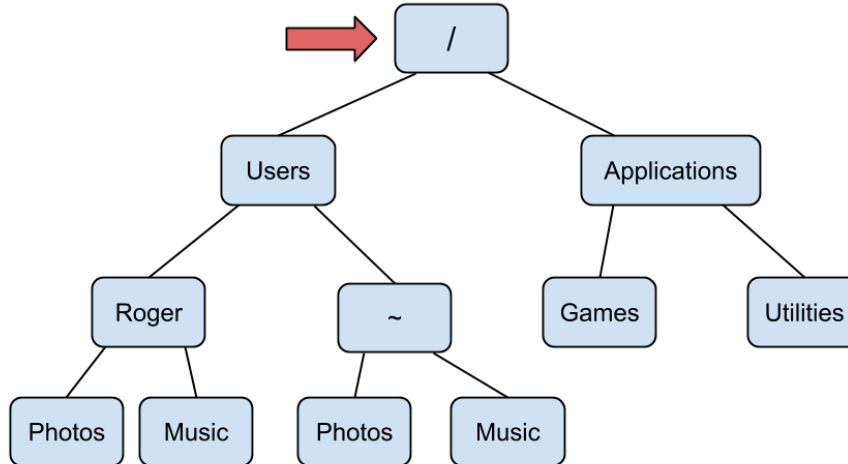
Your computer's directory structure

- The directory structure on your computer looks something like this



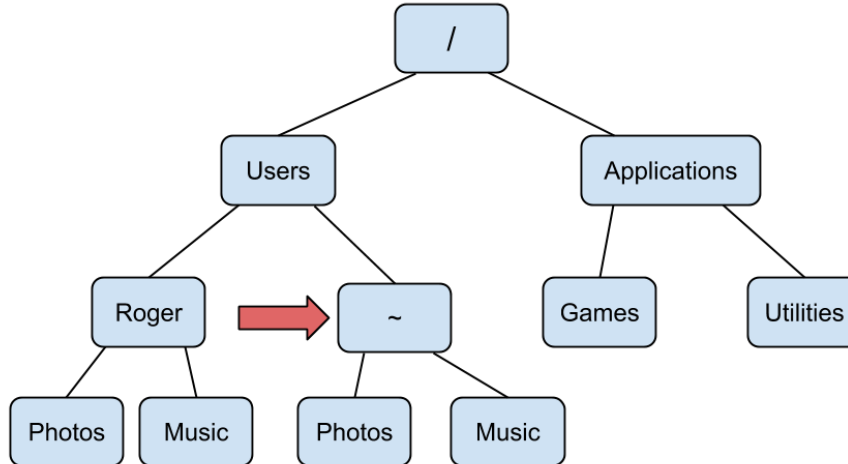
Special directories: root

- The directory at the top of the tree is called the root directory
- The root directory contains all other directories
- The name of this directory is represented by a slash: /



Special directories: home

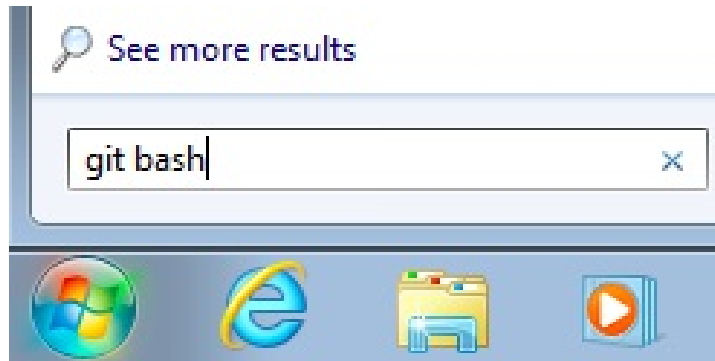
- Your home directory is represented by a tilde: ~
- Your home directory usually contains most of your personal files, pictures, music, etc.
- The name of your home directory is usually the name you use to log into your computer



Navigating directories with the CLI

Windows users:

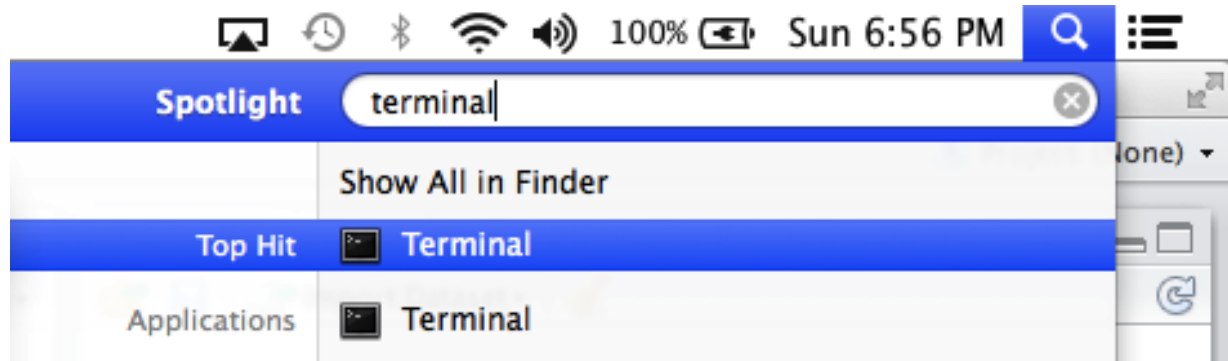
- Open the start menu
- Search for Git Bash
- Open Git Bash



Navigating directories with the CLI

Mac users:

- Open Spotlight
- Search Terminal
- Open Terminal



CLI Basics

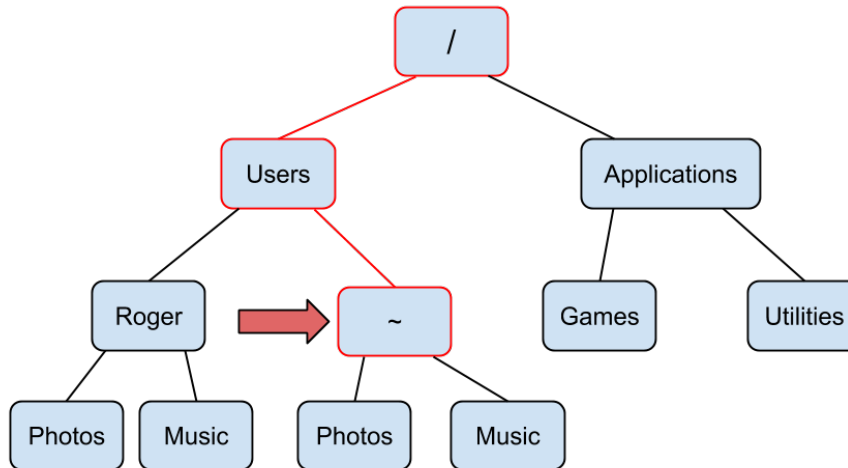
- When you open your CLI you will see your prompt, which will look something like the name of your computer, followed by your username, followed by a \$
- When you open your CLI you start in your home directory.
- Whatever directory you're currently working with in your CLI is called the "working directory"



```
Seans-MacBook-Air:~ sean$
```

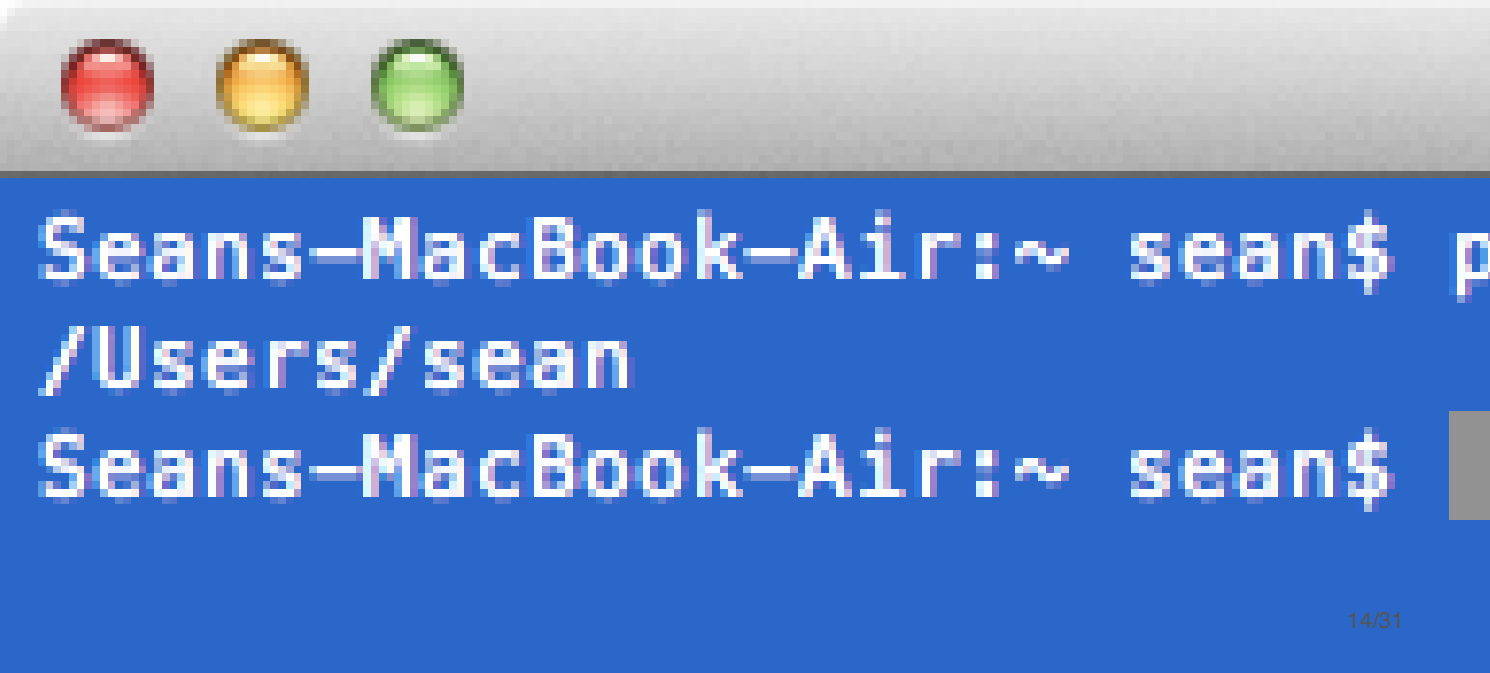
CLI Basics

- You can imagine tracing all of the directories from your root directory to the directory you're currently in.
- This is called the "path" to your working directory.



CLI Basics

- In your CLI prompt, type `pwd` and press enter.
- This will display the path to you're working directory.
- As you can see we get the prompt back after entering a command.

A screenshot of a macOS-style terminal window. The title bar is light gray with three colored window control buttons (red, yellow, green) on the left. The terminal background is blue with white text. The text shows the prompt 'Seans-MacBook-Air:~ sean\$' followed by the command 'pwd' and its output '/Users/sean'. The prompt 'Seans-MacBook-Air:~ sean\$' appears again on the next line.

```
Seans-MacBook-Air:~ sean$ pwd
/Users/sean
Seans-MacBook-Air:~ sean$
```

CLI Commands

- You use the CLI prompt by typing in a command and pressing enter.
- `pwd` can be used at any time to display the path to your working directory (`pwd` is an abbreviation for "print working directory")

CLI Commands

- CLI commands follow this recipe: ***command flags arguments***
- ***command*** is the CLI command which does a specific task
- ***flags*** are options we give to the ***command*** to trigger certain behaviors, preceded by a –
- ***arguments*** can be what the ***command*** is going to modify, or other options for the ***command***
- Depending on the ***command***, there can be zero or more ***flags*** and ***arguments***
- For example `pwd` is a ***command*** that requires no ***flags*** or ***arguments***

CLI Commands

- `pwd` displays the path to the current working directory

```
jeff$ pwd  
/Users/jeff  
jeff$
```

CLI Commands

- `clear` will clear out the commands in your current CLI window

```
jeff$ pwd  
/Users/jeff  
jeff$ clear
```

```
jeff$
```

CLI Commands

- `ls` lists files and folders in the current directory
- `ls -a` lists hidden and unhidden files and folders
- `ls -al` lists details for hidden and unhidden files and folders
- Notice that `-a` and `-l` are flags (they're preceded by a `-`)
- They can be combined into the flag: `-al`

```
jeff$ ls
Desktop  Photos  Music
jeff$ ls -a
Desktop  Photos  Music  .Trash  .DS_Store
jeff$
```

CLI Commands

- `cd` stands for "change directory"
- `cd` takes as an argument the directory you want to visit
- `cd` with no argument takes you to your home directory
- `cd ..` allows you to change directory to one level above your current directory

```
jeff$ cd Music/Debussy
jeff$ pwd
/Users/jeff/Music/Debussy
jeff$ cd ..
jeff$ pwd
/Users/jeff/Music
jeff$ cd
jeff$ pwd
/Users/jeff
jeff$
```

CLI Commands

- `mkdir` stands for "make directory"
- Just like: right click -> create new folder
- `mkdir` takes as an argument the name of the directory you're creating

```
jeff$ mkdir Documents
jeff$ ls
Desktop  Photos  Music   Documents
jeff$ cd Documents
jeff$ pwd
/Users/jeff/Documents
jeff$ cd
jeff$
```

CLI Commands

- `touch` creates an empty file

```
jeff$ touch test_file
jeff$ ls
Desktop  Photos  Music   Documents  test_file
jeff$
```

CLI Commands

- `cp` stands for "copy"
- `cp` takes as its first argument a file, and as its second argument the path to where you want the file to be copied

```
jeff$ cp test_file Documents
jeff$ cd Documents
jeff$ ls
test_file
jeff$ cd ..
jeff$
```

CLI Commands

- `cp` can also be used for copying the contents of directories, but you must use the `-r` flag
- The line: `cp -r Documents More_docs` copies the contents of `Documents` into `More_docs`

```
jeff$ mkdir More_docs
jeff$ cp -r Documents More_docs
jeff$ cd More_docs
jeff$ ls
test_file
jeff$ cd ..
jeff$
```


CLI Commands

- `rm` stands for "remove"
- `rm` takes the name of a file you wish to remove as its argument

```
jeff$ ls
Desktop  Photos  Music   Documents  More_docs  test_file
jeff$ rm test_file
jeff$ ls
Desktop  Photos  Music   Documents  More_docs
jeff$
```

CLI Commands

- You can also use `rm` to delete entire directories and their contents by using the `-r` flag
- **Be very careful when you do this, there is no way to undo an `rm`**

```
jeff$ ls
Desktop  Photos  Music   Documents  More_docs
jeff$ rm -r More_docs
jeff$ ls
Desktop  Photos  Music   Documents
jeff$
```

CLI Commands

- `mv` stands for "move"
- With `mv` you can move files between directories

```
jeff$ touch new_file
jeff$ mv new_file Documents
jeff$ ls
Desktop  Photos  Music   Documents
jeff$ cd Documents
jeff$ ls
test_file  new_file
jeff$
```

CLI Commands

- You can also use `mv` to rename files

```
jeff$ ls
test_file  new_file
jeff$ mv new_file renamed_file
jeff$ ls
test_file renamed_file
jeff$
```

CLI Commands

- `echo` will print whatever arguments you provide

```
jeff$ echo Hello World!  
Hello World!  
jeff$
```

CLI Commands

- `date` will print today's date

```
jeff$ date
```

```
Mon Nov  4 20:48:03 EST 2013
```

```
jeff$
```

Summary of Commands

- `pwd`
- `clear`
- `ls`
- `cd`
- `mkdir`
- `touch`
- `cp`
- `rm`
- `mv`
- `date`
- `echo`