The Terminal

SURP 2022 Python Bootcamp

Ohio State Astronomy

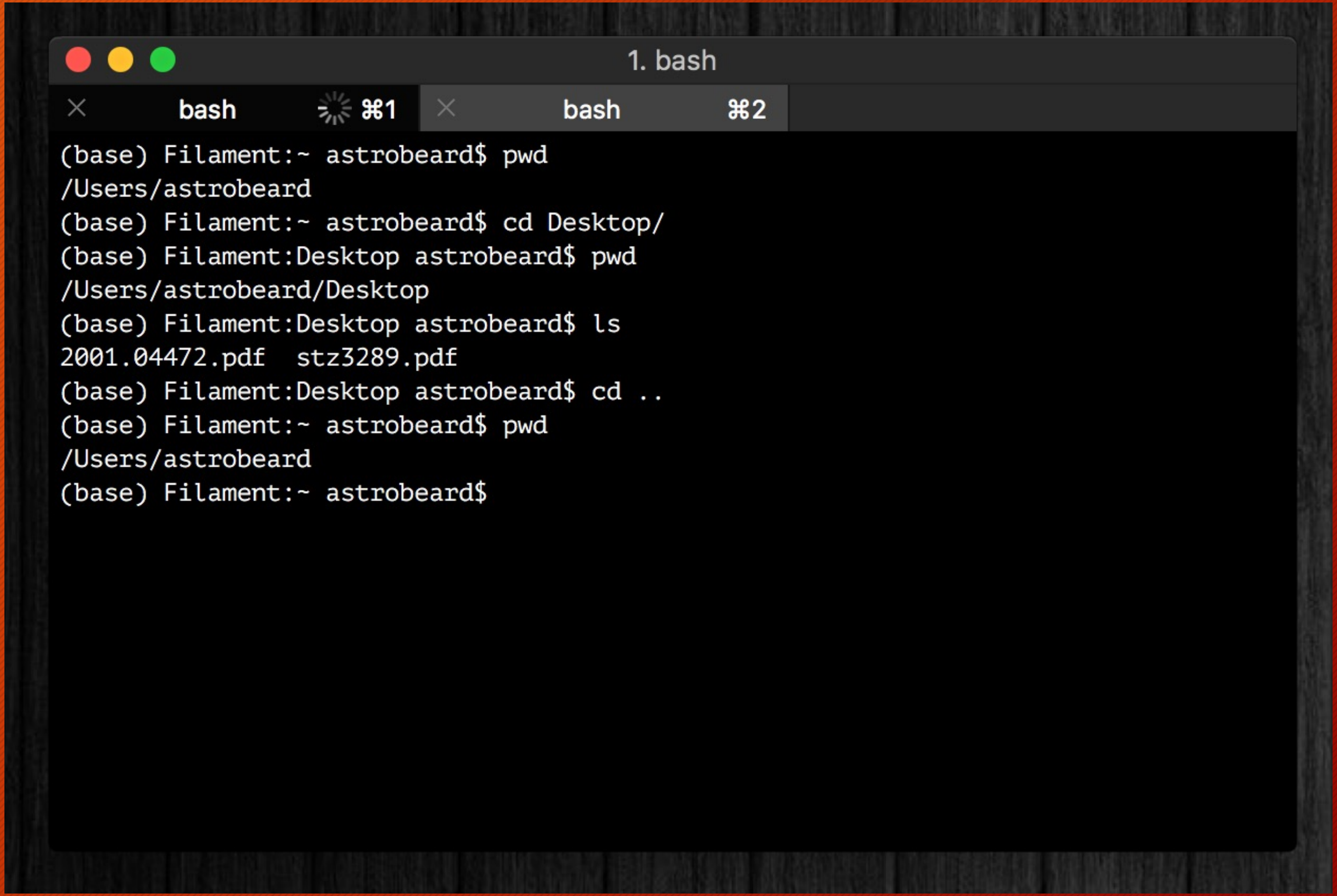
Slides by: James W. Johnson

What is a Terminal?

A command-line interpreter

Executes single commands entered by the user one after another

Think of this as a different interface on a Finder window when learning it



```
(base) Filament:~ astrobeard$ pwd
/Users/astrobeard
(base) Filament:~ astrobeard$ cd Desktop/
(base) Filament:Desktop astrobeard$ pwd
/Users/astrobeard/Desktop
(base) Filament:Desktop astrobeard$ ls
2001.04472.pdf  stz3289.pdf
(base) Filament:Desktop astrobeard$ cd ..
(base) Filament:~ astrobeard$ pwd
/Users/astrobeard
(base) Filament:~ astrobeard$
```

Disclaimer

There are different types of Terminals - the most common one is *bash*, ran in Linux and Mac OS environments

The Windows command-line is NOT a bash environment – to access a bash environment in Windows, you need the Windows Subsystem for Linux (WSL)

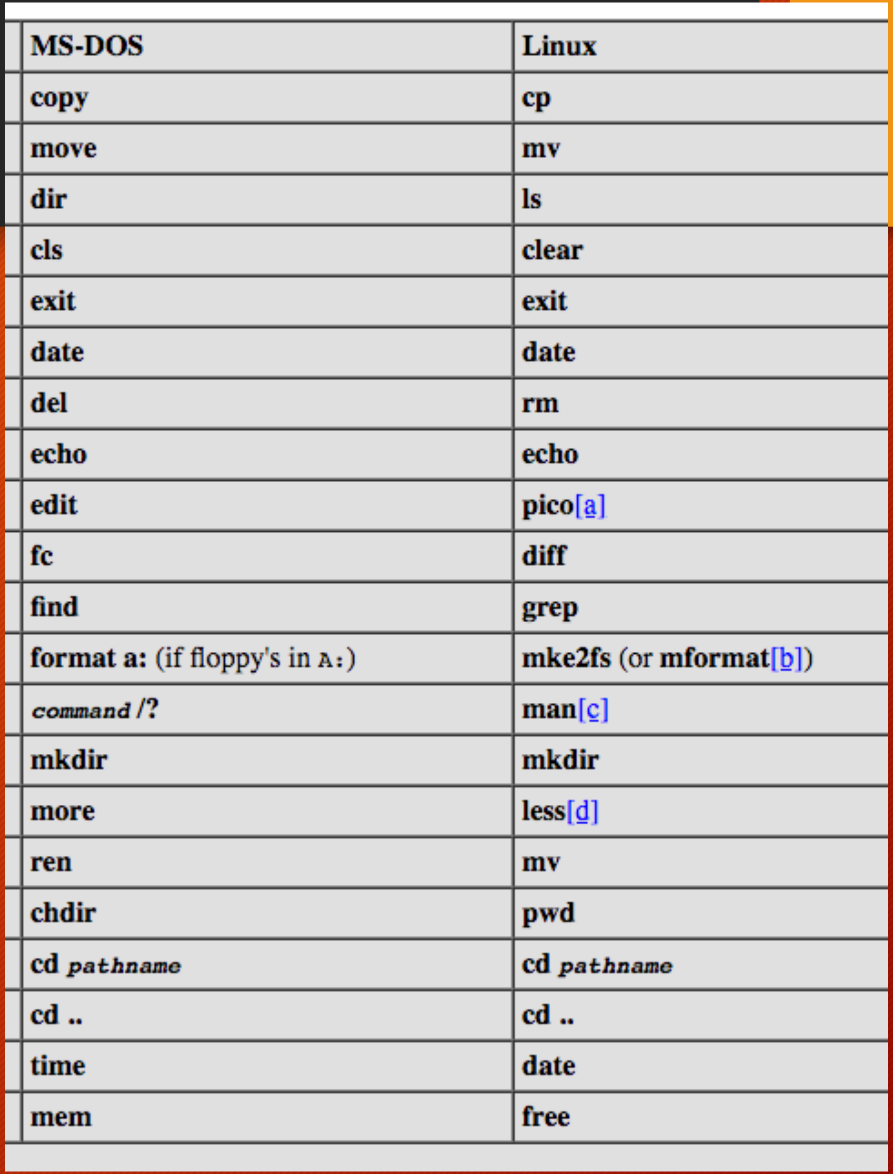
- https://www.microsoft.com/en-us/p/windows-terminal-preview/9n0dx20hk701?activetab=pivot:overviewtab
- Note: This requires Windows 10 version 18362.0 or later
- Another option is use a virtual environment, though these are more data-intensive as they're often running another OS
- Where Unix terminals use $ to reference variables, MS-DOS uses enclosing % symbols

Cheat Sheet

Windows uses the MS-DOS command line system – this cheat sheet will take simple commands from one to the other

https://ftp.kh.edu.tw/Linux/Redhat/en_6.2/doc/gsg/ch-doslinux.htm

There are a wealth of references for using a command line



MS-DOS	Linux
copy	cp
move	mv
dir	ls
cls	clear
exit	exit
date	date
del	rm
echo	echo
edit	pico[a]
fc	diff
find	grep
format a: (if floppy's in A:)	mke2fs (or mformat[b])
command /?	man[c]
mkdir	mkdir
more	less[d]
ren	mv
chdir	pwd
cd *pathname*	cd *pathname*
cd ..	cd ..
time	date
mem	free

echo: Print Statements

Prints a message to the console

Example:

```
$ echo Hello world!
Hello world!
$ echo $x


$ x=3
$ echo $x
3
```

pwd: Print Working Directory [*chdir*]

Prints the name of the directory you’re currently in

Example:

```
$ pwd
/Users/BrutusBuckeye/Desktop/SURP/bootcamp/
```

Note: Windows users should be careful not to confuse this with Python’s *os.chdir* function, whose function is to *change* directories

cd: Change Directory

Change the directory you’re currently in

Example:

```
$ pwd
/Users/BrutusBuckeye
$ cd Desktop/SURP/bootcamp
$ pwd
/Users/BrutusBuckeye/Desktop/SURP/bootcamp
$ cd .. (/Users/BrutusBuckeye/Desktop/SURP)
$ cd ~ (/Users/BrutusBuckeye)
```

ls: List [*dir*]

List all files in a given directory

Example:

```
$ pwd
/Users/BrutusBuckeye/Desktop/SURP/bootcamp
$ ls
exercises notes slides somecode.py
$ cd ..
$ ls
bootcamp plots papers notebook.ipynb textfilecode.py
```

mv: Move [*move*]

Move (i.e. rename) a file or directory

Usage: *mv [old file name] [new file name]*

Example:

```
$ pwd
/Users/BrutusBuckeye/Desktop/SURP/bootcamp
$ mv oldname.py newname.py
$ ls
exercises newname.py notes slides
```

cp: Copy [*copy*]

Copy a file to a new name/location

Usage: *cp [existing file name] [new file name]*

Example:

```
$ ls
data.dat result.out somecode1.py
$ cp result.out copy.out
$ ls
 copy.out data.dat result.out somecode1.py
```

mkdir: Make Directory

Create a new directory (same as clicking “New Folder” in a Finder window)
Usage: *mkdir [directory name]*

Example:

```
$ pwd
/Users/BrutusBuckeye/Desktop/SURP/bootcamp
$ mkdir example
$ ls
example exercises notes slides somecode.py
```

rm: Remove [*del*]

Remove a file from system memory (careful – this doesn’t move a file to trash)
Usage: *rm [filename]*

Example:

```
$ ls
goodcode1.py goodcode2.py badcode.py
$ rm badcode.py
$ ls
goodcode1.py goodcode2.py
```

man: Manual [<*command*>/?]

Pulls up the manual entry (i.e. documentation) for a given terminal command
Can be used as a reference on what "flags" each command takes
Press Q to exit a *man* page

Example:

```
$ man ls [ls/?]
$ man mv [mv/?]
$ man pwd [pwd/?]
```

*: All Files

An asterisk (*) refers to all files in a given directory, and can be modified to refer to only those with a specific prefix or suffix

- Known as *wildcards* or *globs*

Example:

```
$ ls
somedata.dat somecode.py someoutput.out
$ ls *.py
somecode.py
$ ls some*
somedata.dat somecode.py someoutput.out
```

The Bash Profile

A particular file located at ~/.bash_profile (can also use ~/.bashrc)

Typically contains…

- Environment variables
- Aliases
- Modifications to your PATH or PYTHONPATH
- Some gibberish used by *conda*

… if there's even anything there yet

Modifications require running *source ~/.bash_profile* or simply restarting the terminal to take effect

The Bash Profile: Windows Equivalent

No standard name, but files can be set to *autorun* upon terminal start, achieving the same effect

cmd.exe /k "%HOMEDRIVE%\%HOMEPATH%\cmd-startup.bat”

- */k* causes the cmd-startup.bat file to run on launching command line

https://superuser.com/questions/144347/is-there-windows-equivalent-to-the-bashrc-file-in-linux

Disclaimer: If you’re an astronomer, bash is a better choice than PowerShell. This will vary in other fields, but astronomy uses Unix-based operating systems.

Aliases

A way of creating a terminal command out of other terminal commands

Can create one in your terminal independent of your bash profile, but putting them there makes them permanent

Example:

```
alias makeplot=“python plotting_script.py”
alias lc=“ls –lha”
alias surp=“cd ~/Desktop/SURP/”
```

Environment Variables

Variables global to the current shell

Can be created outside the bash profile, but are permanent when put there. Use *export* when adding one to the bash profile

Example:

```
export SURP_DIRECTORY="~/Desktop/SURP/"
```

Can be accessed in Python via os.environ (a dictionary)

PATH and PYTHONPATH

PATH: directories where your computer looks for executables and (more importantly) python code (separated by colons)

PYTHONPATH: additional directories where your computer looks for python code, also separated by colons

Example:

```
export PYTHONPATH=$HOME/path/to/my/python/code:$PYTHONPATH
```

1 million brownie points to whoever knows why $PYTHONPATH appears on the right here

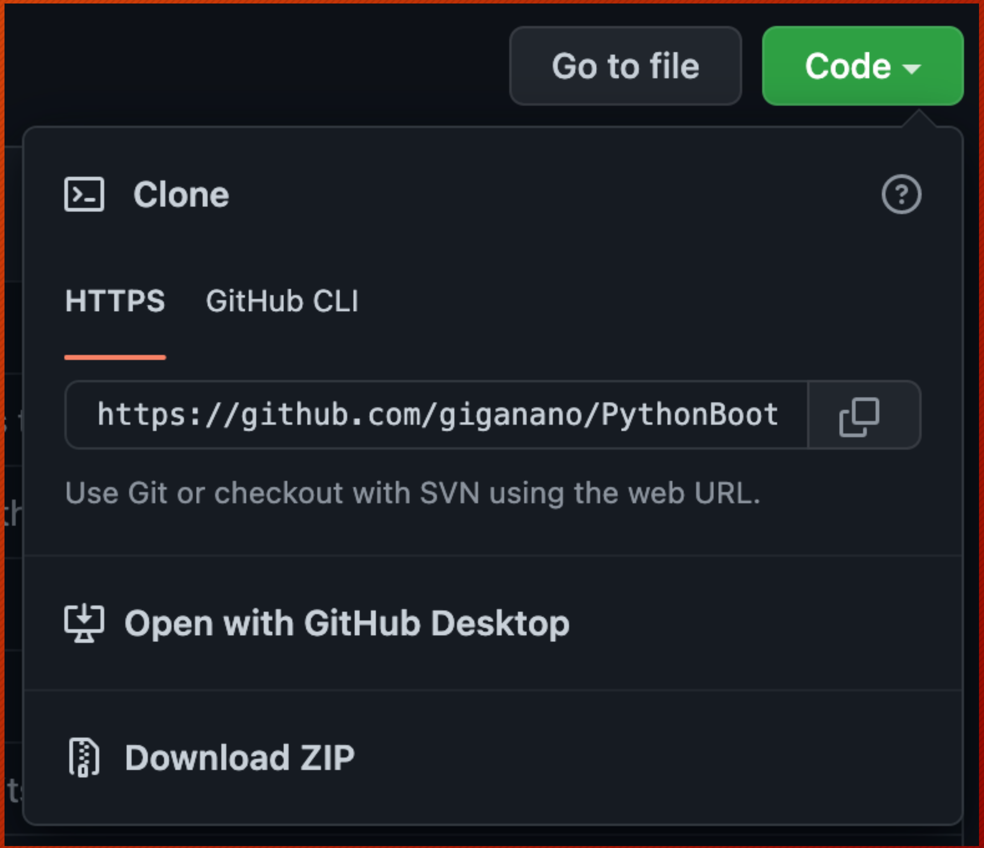
Getting the Bootcamp Material

Online: https://jamesjohnson.space/bootcamp

1. Navigate to the folder you'd like to store it in
2. Run *git clone* *https://github.com/giganano/PythonBootcamp.git*

Or: Download the zip-drive from the same URL

Whenever there are updates: *git pull* from within the bootcamp folder



```
https://github.com/giganano/PythonBoot
```

Use Git or checkout with SVN using the web URL.