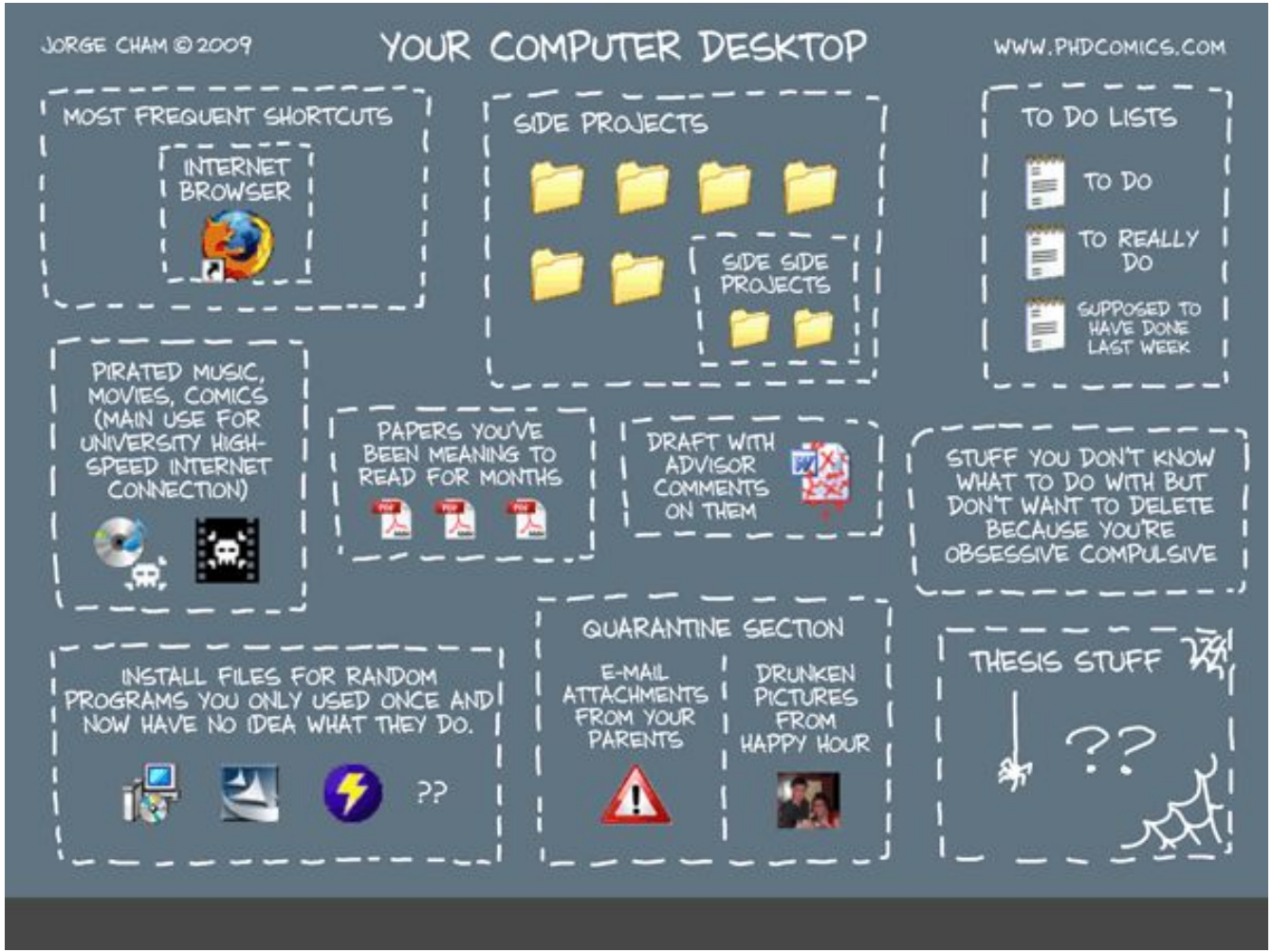
Organizing your research and using bash

Manasi Jayakumar and Arlene Lormestoire
SIPPS 2022 Workshop - June 23, 2022

Why are we talking about organization?

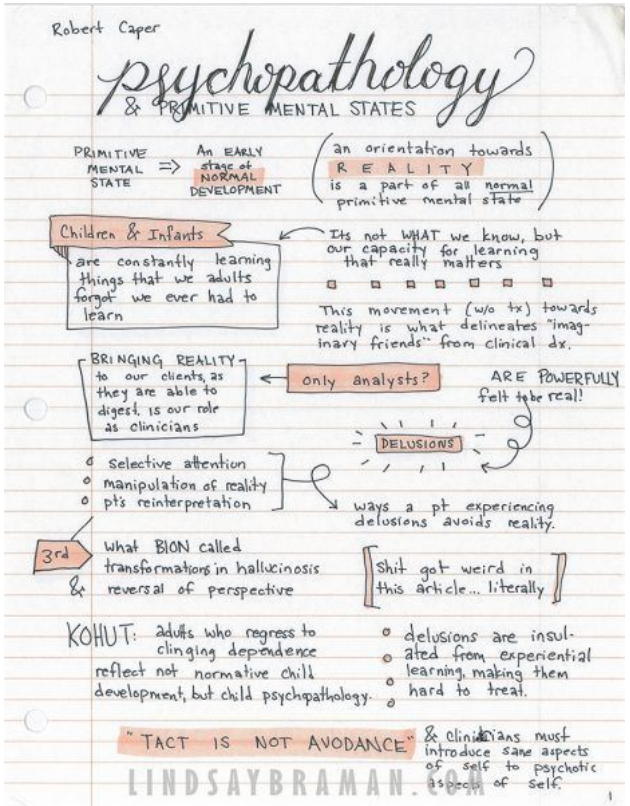
Research doesn’t involve only one type of file, but is usually a collection of files.

Over time becomes a chaotic web!


JORGE CHAM © 2009
YOUR COMPUTER DESKTOP
WWW.PHDCOMICS.COM
MOST FREQUENT SHORTCUTS
INTERNET BROWSER
SIDE PROJECTS
SIDE SIDE PROJECTS
TO DO LISTS
TO DO
TO REALLY DO
SUPPOSED TO HAVE DONE LAST WEEK
PIRATED MUSIC, MOVIES, COMICS (MAIN USE FOR UNIVERSITY HIGH-SPEED INTERNET CONNECTION)
PAPERS YOU'VE BEEN MEANING TO READ FOR MONTHS
DRAFT WITH ADVISOR COMMENTS ON THEM
STUFF YOU DON'T KNOW WHAT TO DO WITH BUT DON'T WANT TO DELETE BECAUSE YOU'RE OBSESSIVE COMPULSIVE
INSTALL FILES FOR RANDOM PROGRAMS YOU ONLY USED ONCE AND NOW HAVE NO IDEA WHAT THEY DO.
??
QUARANTINE SECTION
E-MAIL ATTACHMENTS FROM YOUR PARENTS
DRUNKEN PICTURES FROM HAPPY HOUR
THESIS STUFF
??

Different types of files can be organized in different ways!

Notes



Robert Caper

psychopathology & PRIMITIVE MENTAL STATES

PRIMITIVE MENTAL STATE => An EARLY stage of NORMAL DEVELOPMENT

(an orientation towards REALITY is a part of all normal primitive mental state)

Children & Infants are constantly learning things that we adults forget we ever had to learn

← Its not WHAT we know, but our capacity for learning that really matters

This movement (w/o tx) towards reality is what delineates "imaginary friends" from clinical dx.

BRINGING REALITY to our clients, as they are able to digest, is our role as clinicians ← only analysts?

ARE POWERFULLY felt to be real!

DELUSIONS

- Selective attention
- manipulation of reality
- pt's reinterpretation

→ ways a pt experiencing delusions avoids reality.

3rd & what BION called transformations in hallucinosis reversal of perspective

[Shit got weird in this article... literally]

KOHUT: adults who regress to clinging dependence reflect not normative child development, but child psychopathology.

- delusions are insulated from experiential learning, making them hard to treat.

"TACT IS NOT AVOIDANCE" & Clinicians must introduce sane aspects of self to psychotic aspects of self.

LINDSAYBRAMAN.COM

no trade-off b/n prediction (form) & encoding in the current move.

Q. what does prediction coff reflect?

a) Prediction confidence.

→ prediction strength (how much)

→ prediction ~~e~~entropy (chaotic)

* prediction confidence = prediction strength / prediction entropy *

b) Prediction accuracy.

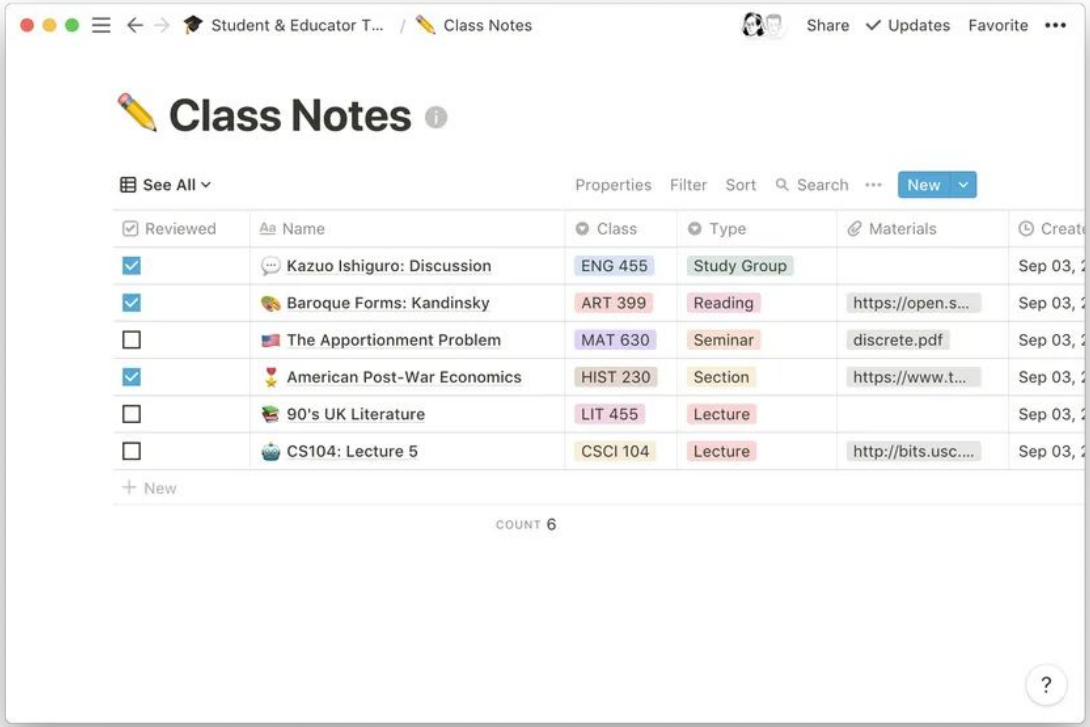
→ time looking at correct next move / time looking (total)

c) Surprise.

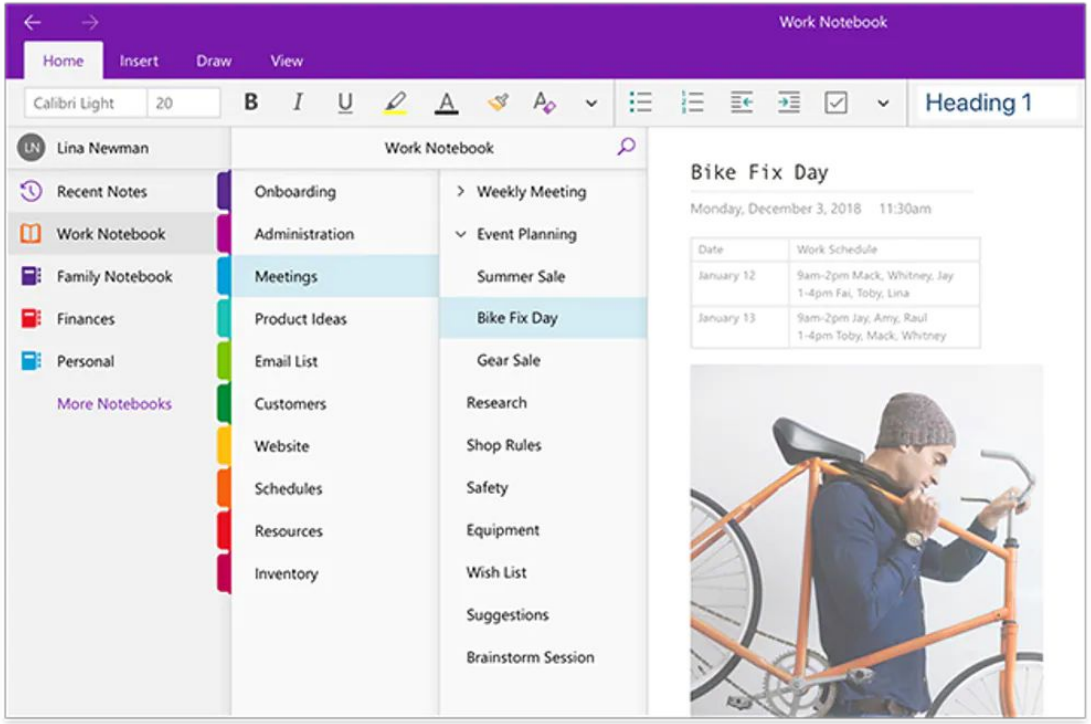
→ -log(prediction accuracy)

Notes

Notion
Pro version is free with .edu email



Notes



Onenote
Comes with microsoft apps, but it is free

Notes

Take notes on EVERYTHING!

Papers

So many ways to read papers…..

Citation managers to store papers

papers

Research Projects

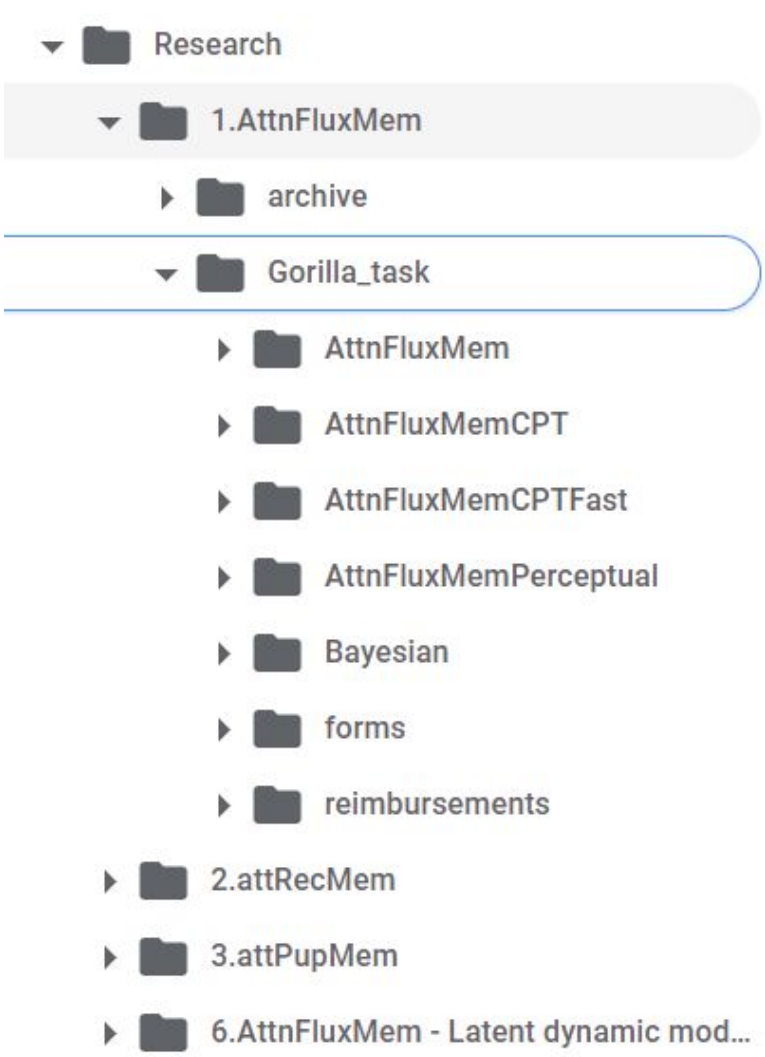
Managing your project

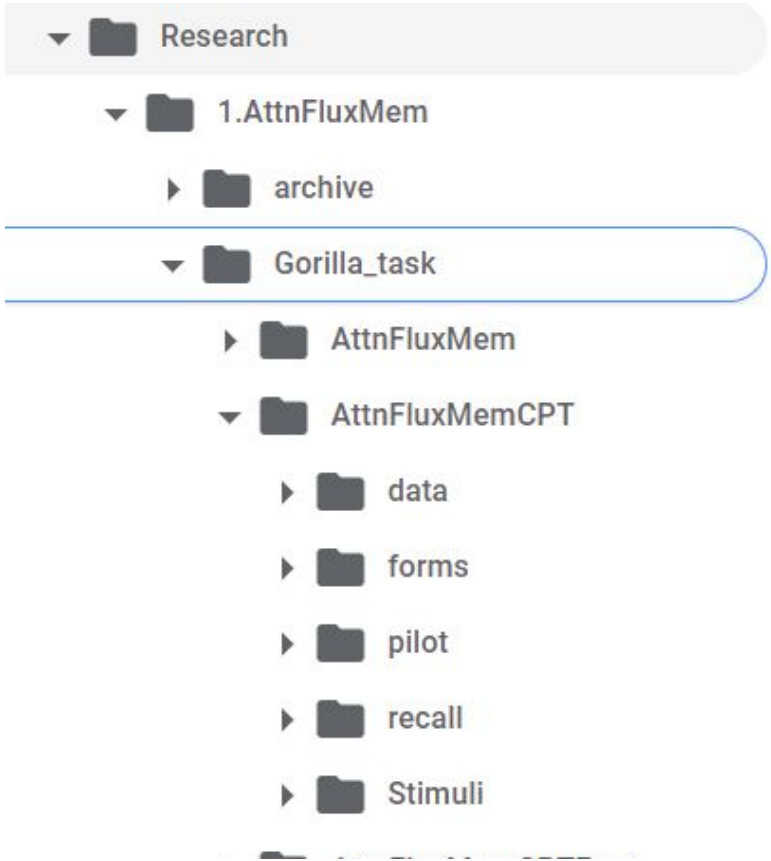
- Whatever you use for your notes
- Specialized apps

Projects - Data and Code

This might be specific to your research, your lab etc. but I think honestly file management and organization for your projects will go a long way!

- Project 1
  - Study 1
    - Pilot
    - Data
    - Stimuli
    - Analysis
      - Pilot code
      - Figures for ppt
  - Study 2
- Project 2
- .....


Research
1.AttnFluxMem
archive
Gorilla_task
AttnFluxMem
AttnFluxMemCPT
AttnFluxMemCPTFast
AttnFluxMemPerceptual
Bayesian
forms
reimbursements
2.attRecMem
3.attPupMem
6.AttnFluxMem - Latent dynamic mod...


Research
1.AttnFluxMem
archive
Gorilla_task
AttnFluxMem
AttnFluxMemCPT
data
forms
pilot
recall
Stimuli

Introduction to the Command Line

What is the command line?

The **command line** is a text-based user interface for your computer. It allows you to type in commands directly to your operating system.

Also referred to as: *prompt, terminal, shell, bash*

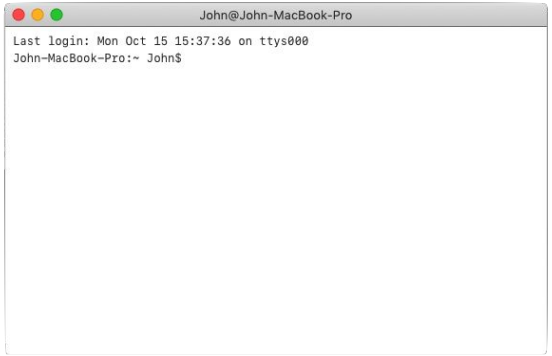
Why use the command line?

- Automate tasks, such as moving/renaming/downloading a large amount of files
- Less resource-intensive
- Helps you develop reproducible data processes

Where is the command line?

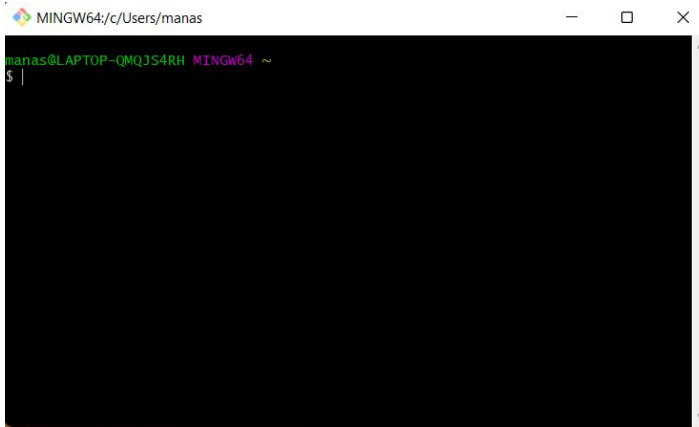
macOS

1. Go to spotlight search (⌘ then space)
2. Type in ‘terminal.app’
3. Launch terminal



Windows

1. Download and Install Git Bash for Windows
2. Press the Windows Start button on the bottom left
3. Type in ‘Git Bash’
4. Click and open Git Bash

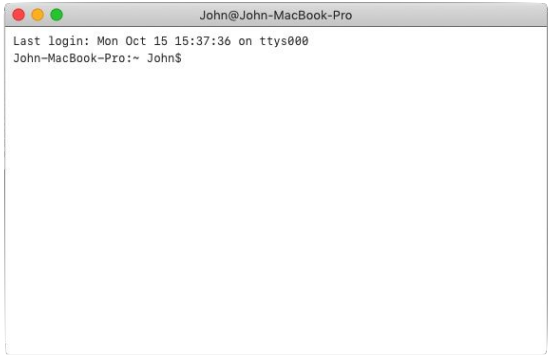


Linux

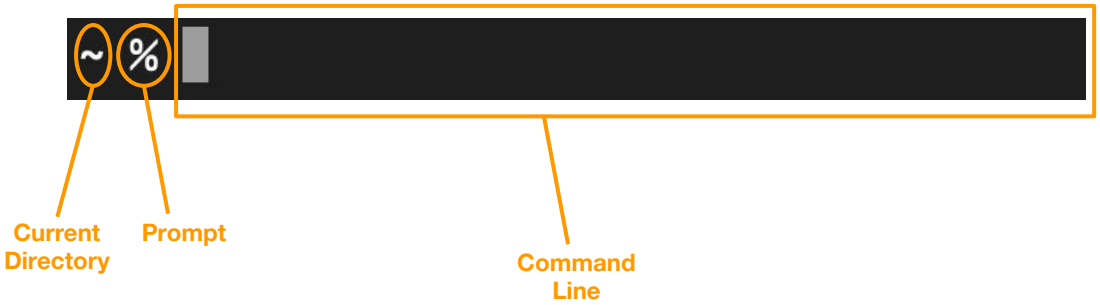
1. Open terminal (ctrl + alt + T)

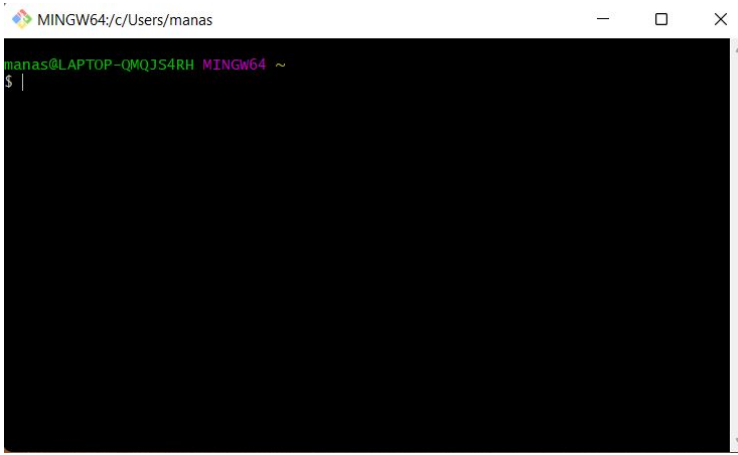
Where is the command line? (cont.)

Upon launching the terminal, the first thing you might see is `$`, `%`, or a combination of your computer name and symbols.



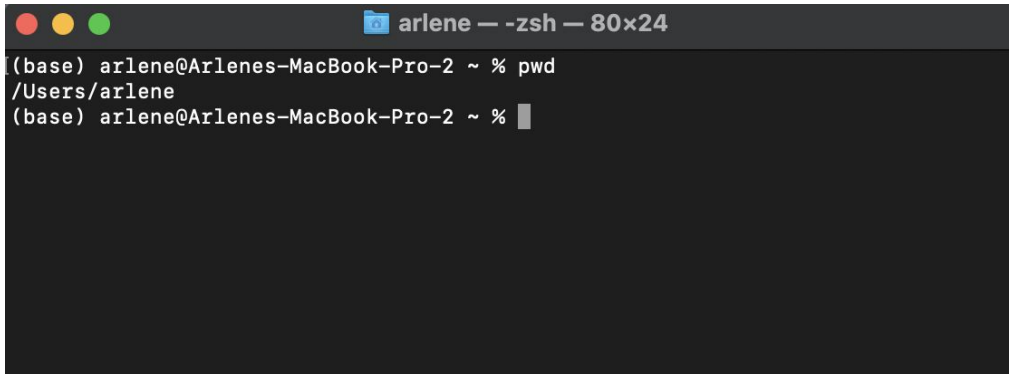
Here's a visual breakdown of what you see in the terminal:





Navigating the Filesystem

Where are we right now?
To figure out your current location, we will use the command `$ pwd`, which stands for **p**rint **w**orking **d**irectory.



This reveals that I am in the `arlene` folder, which is in the `Users` folder on my computer.

Navigating the Filesystem (cont.)

Clearing your window
Before we begin, feel free to use the `clear` command, which will clear your terminal window.

Go to a specific directory
For today's workshop, we will be working out of the `sipps_learnShell` folder, which will be located in the `Downloads` directory.

To 'travel' to a specific directory, we will use the command `cd`, which stands for **c**hange **d**irectory.

In the terminal, type in
`$ cd Downloads/sipps_learnShell`

```
Downloads
|—— sipps_learnShell/
```

Navigating the Filesystem (cont.)

List files in a directory
To list the files in our current directory `Downloads/sipps_learnShell`, we will use **`ls`** which is short for **list**.

In the terminal, type in the command **`$ ls`**

The terminal should print out the directories `folder` and `scripts`, along with the text files `gibberish.txt` and `nonsense.txt`

```
Downloads
|── sipps_learnShell/
|    |── folder/
|    |── gibberish.txt
|    |── nonsense.txt
|    |── scripts/
```

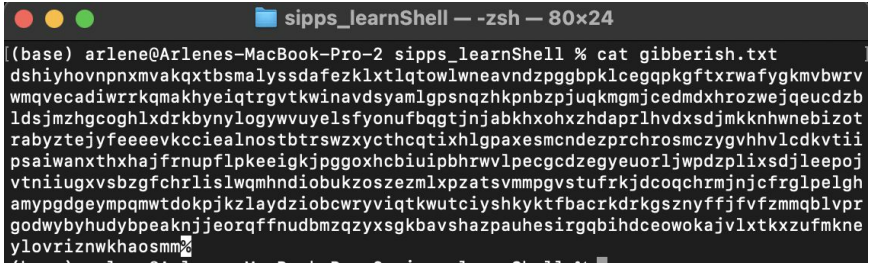
Viewing Files

View a text file

If we are curious about what gibberish.txt contains, we can use the command `cat`

In the terminal, type in `$ cat gibberish.txt`

Tip: You can also type in the first couple of letters, such as "gib" and press tab to have the terminal complete the file name for you!




```
Downloads
|── sipps_learnShell/
|    |── folder/
|    |── gibberish.txt
|    |── nonsense.txt
|    |── scripts/
```

Deleting Files

Deleting Files
Since we don't need this file, we can go ahead and delete it using the command `rm`

Always be very careful while using this command!

In the terminal, type in `$ rm gibberish.txt`

After you do this, then list the contents of the directory by typing `$ ls`. Your file structure should only contain
`folder` and `scripts` now.

```
Downloads
|── sipps_learnShell/
|    |── folder/
|    |── nonsense.txt
|    |── scripts/
```

Renaming folders

Now, let's use **`pwd`** to remind ourselves where we are. We should be in the `sipps_learnShell` directory.

Next, let's see what's in the folder directory by using **`ls`**. In the terminal, type in **`$ ls folder`**

The terminal should print out three files: `sub-01.txt`, `sub-02.txt` and `sub-04.txt`

```
Downloads
|── sipps_learnShell/
|    |── folder/
|    |    |── sub-01.txt
|    |    |── sub-02.txt
|    |    |── sub-04.txt
|    |── scripts/
|    |── nonsense.txt
```

Renaming folders, cont.

Based on its contents, “folder” is too general of a name.
We can change it in the terminal using the command `mv`.
The command `mv` is short for **move**.

Notably, `mv` has two functions:
1) Moving folders
2) **Renaming**

The general structure of `mv` is **mv old_name new_name**

In the terminal, type in `$ mv folder text_files`

```
Downloads
|── sipps_learnShell/
|    |── text_files/
|    |    |── sub-01.txt
|    |    |── sub-02.txt
|    |    |── sub-04.txt
|    |── scripts/
|    |── nonsense.txt
```

Copying files

Next, let's actually go into our newly renamed folder text_files. In the terminal, type in `$ cd text_files`

Let's take a look at one text file for now. Type in `$ cat sub-01.txt`

Now, let's copy what we have from sub-01 to create a file for sub-03, using the `cp` command. This is short for **copy.**
The general structure of cp is: **cp** old_file new_file

In the terminal, type in `$ cp sub-01.txt sub-03.txt`

```
Downloads
|── sipps_learnShell/
|    |── text_files/
|    |    |── sub-01.txt
|    |    |── sub-02.txt
|    |    |── sub-03.txt
|    |    |── sub-04.txt
|    |── scripts/
|    |── nonsense.txt
```

Creating a new directory

Now that we have a full set of text files, let's create a new directory. Let's go back to the `sipps_learnShell` directory. We will use the command cd again.

In terminal, type in **`$ cd ..`** The two dots allow us to go backwards *once*.

In the sipps_learnShell directory, we will create a new directory called "subs" using the command **`mkdir`**

In the terminal, type in **`$ mkdir subs`**

```
Downloads
|── sipps_learnShell/
|    |── text_files/
|    |    |── sub-01.txt
|    |    |── sub-02.txt
|    |    |── sub-03.txt
|    |    |── sub-04.txt
|    |── scripts/
|    |── subs/
|    |── nonsense.txt
```

Creating a new directory (cont.)

In the sipps_learnShell directory, we will create a subject-level directory using the command `mkdir`

Importantly, we want to create two things:

1) A new directory called “subs”
2) Sub-directories for each subject (e.g. subs/sub-01, subs/sub-02)

In the terminal, type in `$ mkdir -p subs/sub-{01..03}`

The `-p` flag allows us to create sub-directories.

```
Downloads
|── sipps_learnShell/
|    |── text_files/
|    |    |── sub-01.txt
|    |    |── sub-02.txt
|    |    |── sub-03.txt
|    |    |── sub-04.txt
|    |── scripts/
|    |── subs/
|    |    |── sub-01/
|    |    |── sub-02/
|    |    |── sub-03/
|    |── nonsense.txt
```

Moving files to a new location

Next, we will move our text files to their respective locations using the command `mv`

We will do this using a quick for loop, executed in one line. Before we do, let's break this down into three parts:

```
1.  for s in {01..03}
2.  do mv text_files/sub-$s.txt subs/sub-$s
3.  done
```

The full command is the following:

```
$ for s in {01..03}; do mv text_files/sub-$s.txt subs/sub-$s; done
```

Using grep

`grep` allows you to search a text file for a given string. This command can print line numbers of the string, which is helpful for large files.

We can search our text files to see whether the string “September” is included.

In terminal, you can type in the following command:
`grep -r -n September *`

The `-r` flag checks the current directory *along with subdirectories*. The `-n` flag prints the line number that contains our string of interest. The `*` specifies the files in the current directory.

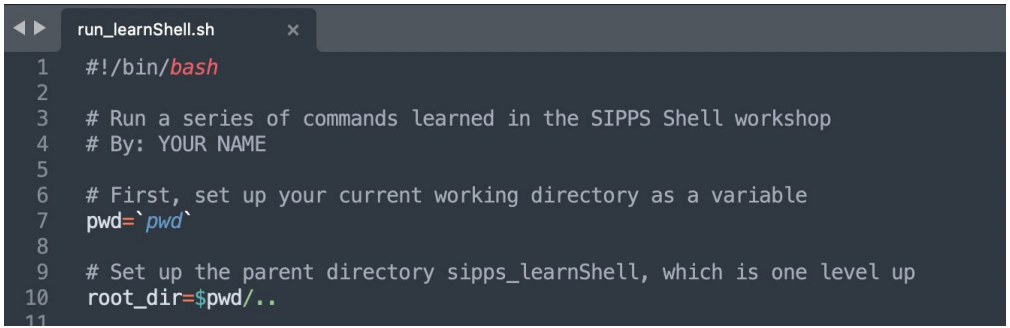
```
Downloads
|── sipps_learnShell/
|    |── text_files/
|    |    |── sub-04.txt
|    |── scripts/
|    |── subs/
|    |    |── sub-01/
|    |    |    |── |── sub-01.txt
|    |    |── sub-02/
|    |    |    |── |── sub-02.txt
|    |    |── sub-03/
|    |    |    |── |── sub-03.txt
|    |── nonsense.txt
```

Using a shell script

Although we have made changes today directly in the terminal, we can also make all of these changes *using one line* (technically).

A **shell script** is text file that contains a series of commands that can be ran by the shell. Today, we will be using `scripts/run_learnShell.sh`

Upon opening our script in our chosen code editor, we get a glimpse of the general structure of a bash script →



Using a shell script (cont.)

In order to run our shell script, we need to make sure that we are in our `scripts` directory. In the terminal, enter the scripts folder by typing in **`$ cd scripts`**

In order to launch the shell script, we will use the following command: **`$ ./run_learnShell.sh`**

```
Downloads
|── sipps_learnShell/
|   |── text_files/
|   |   |── sub-04.txt
|   |── scripts/
|   |   |── run_learnShell.sh
|   |── subs/
|   |   |── …
|   |── nonsense.txt
```