

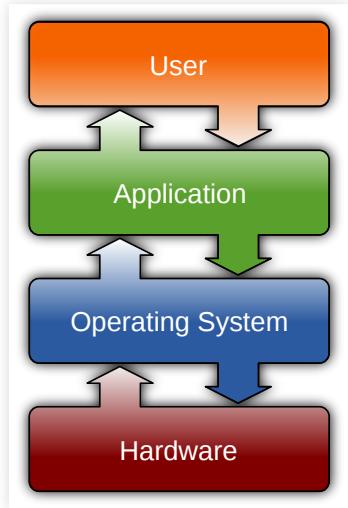
# **STORYTELLING WITH DATA**

**JANUARY 22, 2018**

**GETTING ACQUAINTED WITH OUR  
ANALYTIC ENVIRONMENT**

# **WHAT IS AN OPERATING SYSTEM (OS)?**

# A BRIDGE BETWEEN APPLICATIONS AND HARDWARE



The most common OSs are [Windows](#) and [macOS](#).

The OS should not be confused with the user interface, be it a terminal or GUI.

# GRAPHICAL USER INTERFACES ARE QUITE POPULAR





# AN OS GUI IS JUST A SPECIAL TYPE OF **SHELL**

Text-based shells (e.g. the [Windows command prompt](#) or [Bash](#)) can present as intimidating, but they are your best friend. Graphical shells are just wrappers around some of the commands that shells can execute.







# **WHAT IS A VIRTUAL MACHINE?**

# APPLICATIONS THAT EMULATE SOFTWARE SYSTEMS

- VMs allow you to try another OS without a full install
- VMs can enable the use of applications that cannot be run on some systems
- VMs can help us ensure that all of our programs run the same way

# WHY ARE WE USING VIRTUAL MACHINES?

- When software is developed, it assumes a given method of locating the resources it needs
- Typically, software is developed in a single environment, and then adapted to work elsewhere
- Adaptions can create complications in application behavior



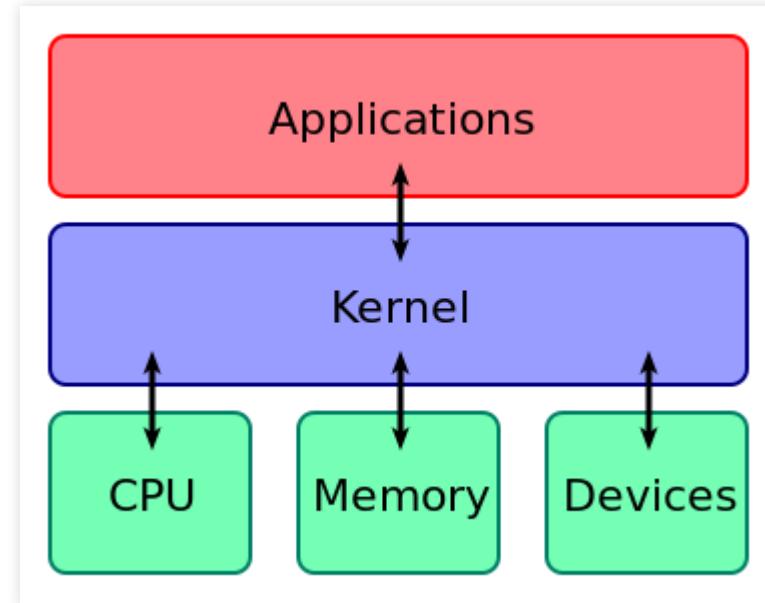
# WHAT IS LINUX?

Freedom. Choices. Beautiful.



# LINUX IS NOT AN OS. IT'S A KERNEL

Developed by Linus Torvalds,  
Linux is a catchall name for an  
open-source kernel.



# A KERNEL IS ONLY ONE PART OF THE OS...

...but it is the most important part. All in, an OS provides a **variety of functions**:

- Loads and manages processes
- Provides interface to hardware
- Provides a file system
- Provides a user interface

The kernel essentially coordinates these tasks.



# WHY DO WE CARE AGAIN?

- Linux distributions tend to be free for personal use
- Linux distributions tend to be open source
- Linux distributions tend to come packaged with many useful development tools and libraries
- For all these reasons, analytic packages are often designed and tested for use on Linux systems







**ENTER UBUNTU...**

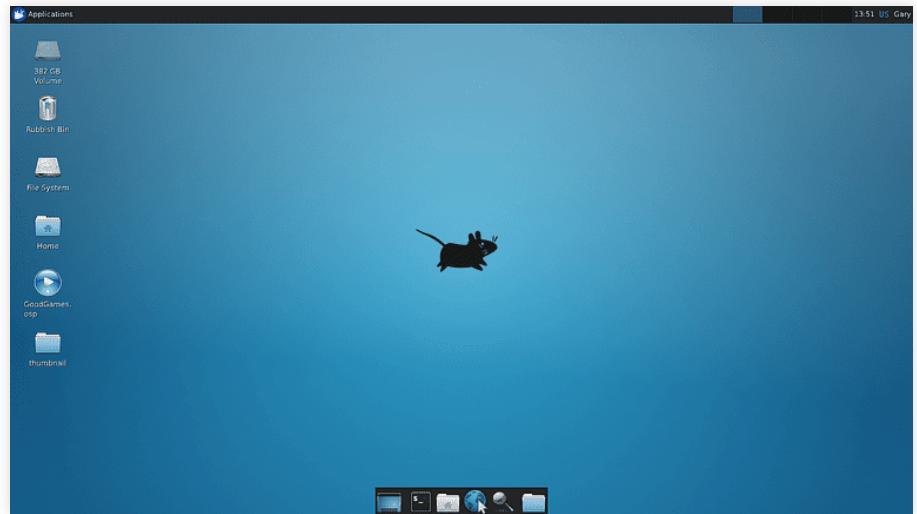


# **UBUNTU IS ONE OF THE MOST POPULAR LINUX DISTRIBUTIONS BECAUSE...**

- ...the GUI is intuitive and easy to use;
- ...it provides full shell functionality
- ...it makes it easy to install and run programs

# UBUNTU IS AN OS, NOT A GUI

Multiple command line and graphical shells are compatible.  
We will be using [xfce](#), but [gnome](#) is another popular choice, and [unity](#) used to ship with the default Ubuntu install.





# **HOW DO I USE A LINUX DISTRIBUTION LIKE UBUNTU?**

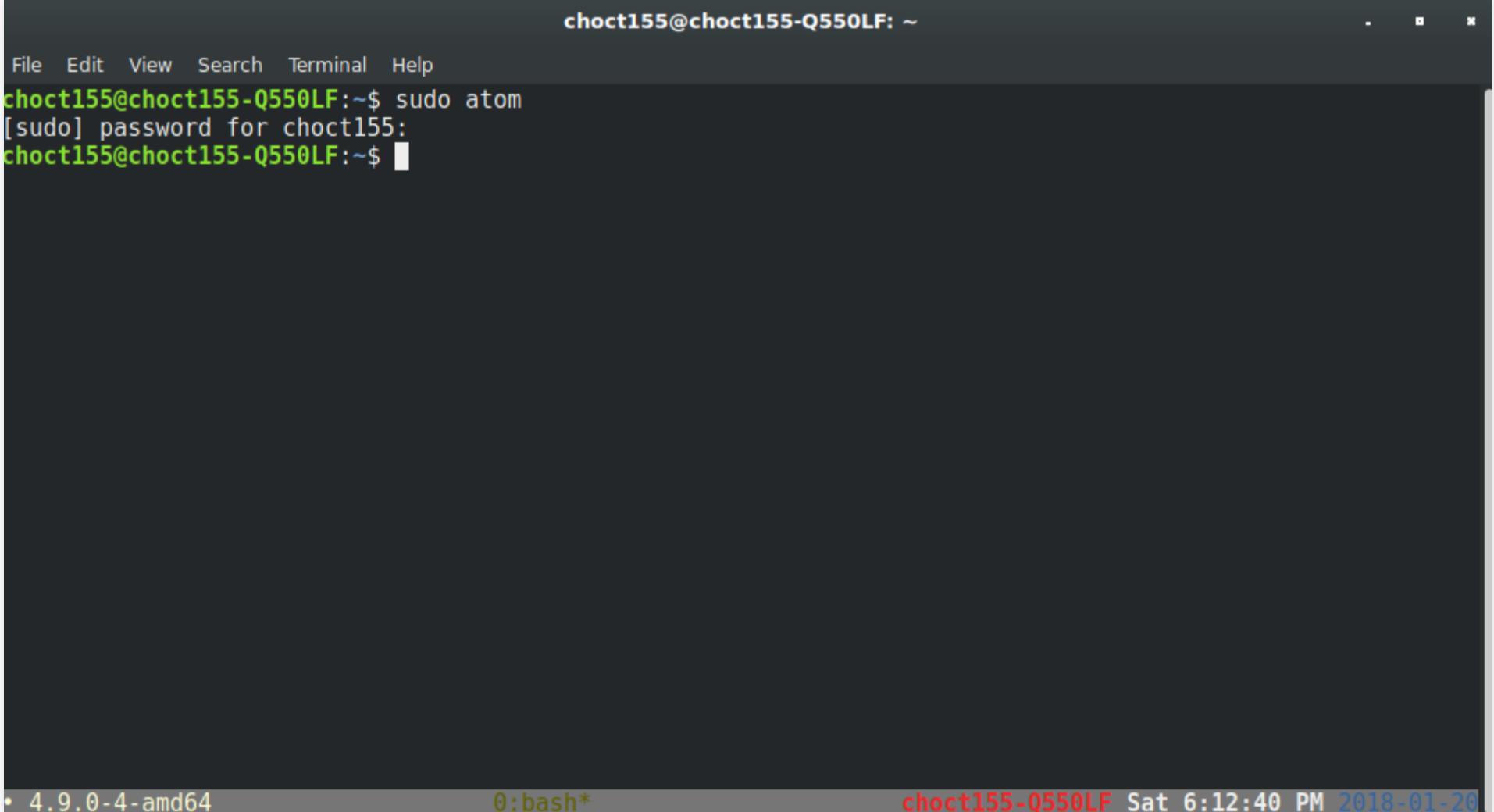
**(SPOILER: YOU CAN USE IT JUST LIKE WINDOWS OR  
MACOS)**

# SUPPOSE I WANT TO OPEN AN EDITOR LIKE **ATOM**

You can navigate to the Atom Editor icon and simply double-click on it...



# ... OR YOU CAN USE YOUR FRIEND, THE SHELL



A screenshot of a terminal window with a dark background. At the top, there's a menu bar with options: File, Edit, View, Search, Terminal, and Help. The title bar shows the user's name and computer: choct155@choct155-Q550LF: ~. In the main window area, the command `sudo atom` is typed, followed by the prompt [sudo] password for choct155: A password entry field is visible. The bottom of the window shows the system information: 4.9.0-4-amd64, 0:bash\*, and the date/time: choct155-Q550LF Sat 6:12:40 PM 2018-01-20.

```
choct155@choct155-Q550LF: ~
File Edit View Search Terminal Help
choct155@choct155-Q550LF:~$ sudo atom
[sudo] password for choct155:
choct155@choct155-Q550LF:~$ █
```

• 4.9.0-4-amd64 0:bash\* choct155-Q550LF Sat 6:12:40 PM 2018-01-20



**THEN YOU CAN CODE TO YOUR  
HEART'S CONTENT.**

The screenshot shows a terminal window with a file explorer sidebar on the left. The sidebar lists files and folders for a project named 'dockra'. The files listed are: resources/osx, src, .babelrc, .eslintrc, .gitignore, gulpfile.babel.js (which is currently open in the main editor), package.json, README.md, and webpack.config.js.

The main area displays the content of the 'gulpfile.babel.js' file. The code defines three Gulp tasks: 'css', 'javascript', and 'html'. The 'css' task concatenates 'main.css' and minifies it. The 'javascript' task uses plumber, sourcemaps, babel, and uglify. The 'html' task is partially defined. The file ends with a comment indicating the path to the Gulp executable.

```
46     'node_modules/' ->
47     ] ->
48   }).on('error', $.sass.logError)) ->
49   .pipe($.concat('main.css')) ->
50   .pipe($.minifyCss()) ->
51   .pipe($.sourcemaps.write()) ->
52   .pipe(gulp.dest('build/dev')) ->
53 );
54
55 gulp.task('javascript', () => {
56   return gulp.src(javascriptFiles) ->
57     .pipe($.plumber()) ->
58     .pipe($.sourcemaps.init()) ->
59     .pipe($.babel()) ->
60     .pipe($.uglify()) ->
61     .pipe($.sourcemaps.write()) ->
62     .pipe(gulp.dest('build/dev')) ->
63 );
64
65 // gulp.task('html', () => {
```

Below the code, the terminal output shows the execution of the command '/Users/noseglid/devel/dockra/node\_modules/.bin/gulp build'. The log indicates the start of each task ('Starting'), its completion ('Finished'), and the total duration for each task.

```
/Users/noseglid/devel/dockra/node_modules/.bin/gulp build
[21:22:20] Using gulpfile ~/devel/dockra/gulpfile.babel.js
[21:22:20] Starting 'css'...
[21:22:21] Starting 'javascript'...
[21:22:21] Starting 'html'...
[21:22:21] Starting 'images'...
[21:22:21] Starting 'fonts'...
[21:22:22] Finished 'html' after 1.34 s
[21:22:22] Finished 'images' after 1.34 s
[21:22:24] Finished 'css' after 3.37 s
```

The bottom of the terminal window shows various status indicators: Line 0, File 0, Project 0, gulpfile.babel.js, Gulp: build, 72:34, LF, Normal, UTF-8, JavaScript, and master.





**BASH IS PROBABLY THE MOST  
UBQUITOUS SHELL IN COMMON  
USAGE ON UNIX SYSTEMS**



# THE BEST WAY TO LEARN ABOUT IT IS BY USING IT

Bash is a low-level interface that allows the user to issue commands to the computer at the command line or by executing a script. Features include but are not limited to ...

- ...potentially complex file management
- ...permissions control
- ...process management
- ...application installation and control

You will be using Bash to, at the very least, start up your VM and version control your projects

# Linux Bash Shell Cheat Sheet

## Basic Commands

### Basic Terminal Shortcuts

```
CTRL L = Clear the terminal  
CTRL D = Logout  
SHIFT Page Up/Down = Go up/down the terminal  
CTRL A = Cursor to start of line  
CTRL E = Cursor the end of line  
CTRL U = Delete left of the cursor  
CTRL K = Delete right of the cursor  
CTRL W = Delete word on the left  
CTRL Y = Paste (after CTRL U,K or W)  
TAB = auto completion of file or command  
CTRL R = reverse search history  
!! = repeat last command  
CTRL Z = stops the current command (resume with fg in foreground or bg in background)
```

### Basic Terminal Navigation

```
ls -a = list all files and folders  
ls <folderName> = list files in folder  
ls -lh = Detailed list, Human readable  
ls -l *.jpg = list jpeg files only  
ls -lh <fileName> = Result for file only  
  
cd <folderName> = change directory  
    if folder name has spaces use “ ”  
cd / = go to root  
cd .. = go up one folder, tip: ../../..  
  
du -h: Disk usage of folders, human readable  
du -ah: “ “ “ files & folders, Human readable  
du -sh: only show disc usage of folders  
  
pwd = print working directory  
  
man <command> = shows manual (RTFM)
```

### Basic file manipulation

```
cat <fileName> = show content of file  
    (less, more)  
head = from the top  
    -n <#oflines> <fileName>  
  
tail = from the bottom  
    -n <#oflines> <fileName>  
  
mkdir = create new folder  
mkdir myStuff ..  
mkdir myStuff/pictures/ ..  
  
cp image.jpg newimage.jpg = copy and rename a file  
cp image.jpg <folderName>/ = copy to folder  
cp image.jpg folder/sameImageNewName.jpg  
cp -R stuff otherStuff = copy and rename a folder  
cp *.txt stuff/ = copy all of *<file type> to folder  
  
mv file.txt Documents/ = move file to a folder  
mv <folderName> <folderName2> = move folder in folder  
mv filename.txt filename2.txt = rename file  
mv <fileName> stuff/newfileName  
mv <folderName>/ .. = move folder up in hierarchy  
  
rm <fileName> .. = delete file (s)  
rm -i <fileName> .. = ask for confirmation each file  
rm -f <fileName> = force deletion of a file  
rm -r <foldername>/ = delete folder  
  
touch <fileName> = create or update a file  
  
ln file1 file2 = physical link  
ln -s file1 file2 = symbolic link
```



# PYTHON IS A GENERAL PURPOSE COMPUTING LANGUAGE

The screenshot shows the Python.org homepage. At the top left is the Python logo and the word "python™". To its right is a search bar with a magnifying glass icon, a "GO" button, and a "Socialize" button. Below the header is a navigation bar with links for "About", "Downloads", "Documentation", "Community", "Success Stories", "News", and "Events".

In the main content area, there is a code editor window containing Python code to generate a Fibonacci series up to n. The code is as follows:

```
# Python 3: Fibonacci series up to n
>>> def fib(n):
>>>     a, b = 0, 1
>>>     while a < n:
>>>         print(a, end=' ')
>>>         a, b = b, a+b
>>>     print()
>>> fib(1000)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610
987
```

To the right of the code editor is a sidebar with the heading "Functions Defined". It contains text explaining the core of extensible programming: defining functions. It mentions mandatory and optional arguments, keyword arguments, and arbitrary argument lists, with a link to "More about defining functions in Python 3". Below this are five numbered buttons (1, 2, 3, 4, 5) for navigating through the page.

At the bottom of the main content area, there is a promotional message: "Python is a programming language that lets you work quickly and integrate systems more effectively. [» Learn More](#)".

**WE WILL LEAN HEAVILY ON THE  
ANACONDA DISTRIBUTION**

## ANACONDA NAVIGATOR

Desktop Portal to Data Science

## ANACONDA PROJECT

Portable Data Science Encapsulation

## DATA SCIENCE LIBRARIES

Data Science IDEs



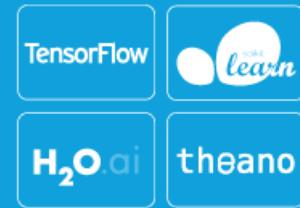
Analytics &amp; Scientific Computing



Visualization



Machine Learning



...and many more!

**CONDA<sup>®</sup>**

Data Science Package &amp; Environment Manager

# THE BIGGEST ADVANTAGE OF ANACONDA IS THE CONDA PACKAGE MANAGER

- conda runs on Windows, macOS, and Linux
- It helps find and install packages
- It manages dependencies across packages
- It makes it easy to manage package environments





THE ANACONDA PROJECT IS ALSO  
COUPLED TIGHTLY WITH THE  
**JUPYTER PROJECT**



Project Jupyter exists to develop open-source software, open-standards, and services for interactive computing across dozens of programming languages.









# OUR CODING ENVIRONMENT: JUPYTER LAB

The screenshot shows the Jupyter Lab interface with several components:

- File Explorer (Left):** Shows a list of files and notebooks, including "MNIST.ipynb", "convnets.ipynb" (selected), "timer.py", and "README.md".
- Code Editor (Top Right):** Displays the content of "timer.py".
- Notebook Cells (Center):** Two cells are visible:
  - In [47]:

```
train_hashes = set(hash(i.tostring()) for i in train_dataset)
test_hashes = set(hash(i.tostring()) for i in test_dataset)
valid_hashes = set(hash(i.tostring()) for i in valid_dataset)
```
  - In [66]:

```
with Timer('28x 28 Pixel Photo Set Sizes {} {}'.format(
    len(train_dataset), len(test_dataset), len(valid_dataset))):
    train_hashes = set(hash(i.tostring()) for i in train_dataset)
    test_hashes = set(hash(i.tostring()) for i in test_dataset)
    valid_hashes = set(hash(i.tostring()) for i in valid_dataset)

    result = np.ndarray(shape=(3, 3))
    hashes = [train_hashes, test_hashes, valid_hashes]

    for i, first_hashes in enumerate(hashes):
        for j, next_hashes in enumerate(hashes):
            result[i, j] = 100.0 * len(first_hashes.intersection(next_hashes)) / len(first_hashes)

    ax = sns.heatmap(result, annot=True)
    plt.title('Percentage overlap between sets (with floating point error)')
    ax.set_yticklabels(['Train', 'Test', 'Valid'])
    ax.set_xticklabels(['Train', 'Test', 'Valid'])
    plt.show();
```
- Figure (Bottom Center):** A heatmap titled "Percentage overlap between sets (with floating point error)" showing the overlap between Train, Test, and Valid datasets.
- Terminal (Bottom Left):** Shows a command-line session in a terminal window titled ".mpies/udacit".























# JUPYTER LAB OFFERS MULTIPLE WAYS TO USE PYTHON

A Python interpreter can always be started from the command line with `python` or `ipython`. Jupyter Lab provides more options:

- [Qt Console](#) is an enhanced interpreter that allows for inline charting
- [Notebook](#) is a web-based application that allows inline charting, embedding of links and other resources, and the flexibility of web development
- Since a shell is available inside of JLab, you can also execute Python [scripts](#) that you have written in an editor tab.









































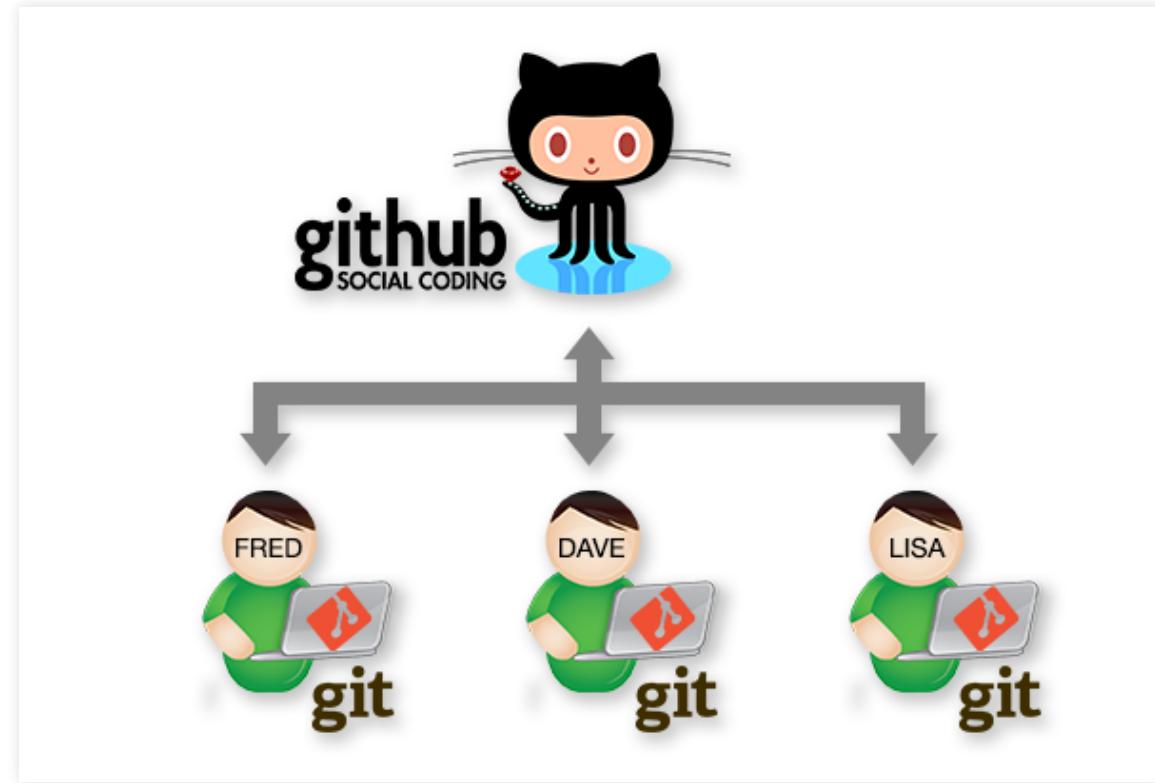






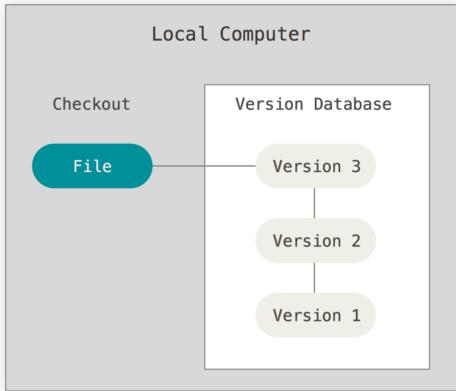
**TO CODE IS GOOD, TO REPLICATE  
DIVINE**





# **VERSION CONTROL IS NOT A SILVER BULLET...**

...but it is a necessary component of reproducible research



It enables you to code without fear of mistake, because you can always go back to an earlier version



# GITHUB WILL BE OUR COORDINATING MECHANISM



- You will need to [create a Github account](#)
- Github has a great collection of [guides](#) for common tasks
- A solid [cheat sheet](#) of common commands is available
- For more than you ever wanted to know, check out [Pro Git](#)







**DATA FILES CAN COME IN A WIDE  
VARIETY OF FORMATS**

# THEY DO NOT ALWAYS LOOK LIKE TABLES!

There is a big world out there beyond Excel:

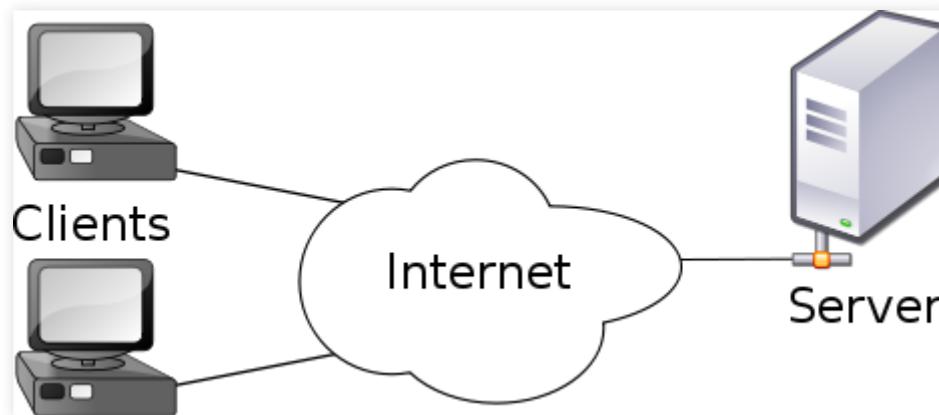
- Tabular data (e.g. CSV, TSV, fixed width)
- Non-tabular data (e.g. JSON)
- Spatial data (e.g. SHP, GeoJSON)
- Relational database data (e.g. DB)

# **SOMETIMES DATA WILL NOT COME FROM LOCALLY STORED FILES**

- We can simulate data via random draws from a statistical distribution
- We can hard code data
- We can access data from remote server

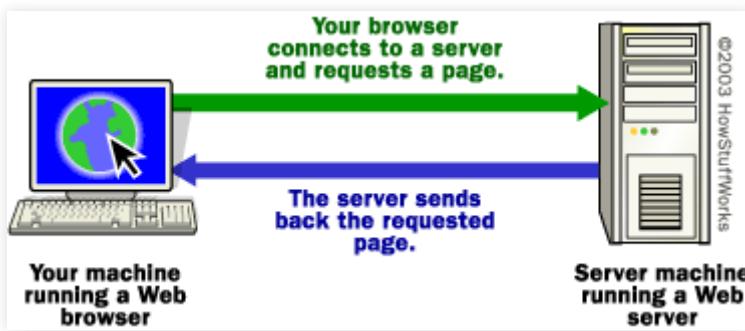


# WHAT'S ALL THIS ABOUT REMOTE SERVERS??



# ANY TIME YOU VISIT A WEBSITE, YOU ARE USING A REMOTE SERVER

A server is just another computer that you have connected to that holds the content you want.



It turns out data can be accessed in a very similar way with the right Uniform Resource Locator (URL)

# ENTER DATA APPLICATION PROGRAMMING INTERFACES...

An API, in general, is just a clearly defined set of methods that allow software components to communicate with each other. In the data realm, APIs help our machine communicate with a remote server via [URL](#), so we can retrieve data from it.

