PY410 / 505
Computational Physics 1

Salvatore Rappoccio

Technical Lectures

- We will have several technical lectures to get you up to speed with programming and technical skills
  - Using UNIX/LINUX command line
  - Compiling, version control
  - Data representations, flow of control, object oriented programming, simple algorithms

Executing code

- The official environment for the class is a container:
  - Docker image for class software (Ubuntu based):
    - https://hub.docker.com/r/ubsuny/compphys

- Software for this class:
  - https://github.com/ubsuny/CompPhys

- If you want to use your laptop environment directly, you can but CAVEAT EMPTOR.

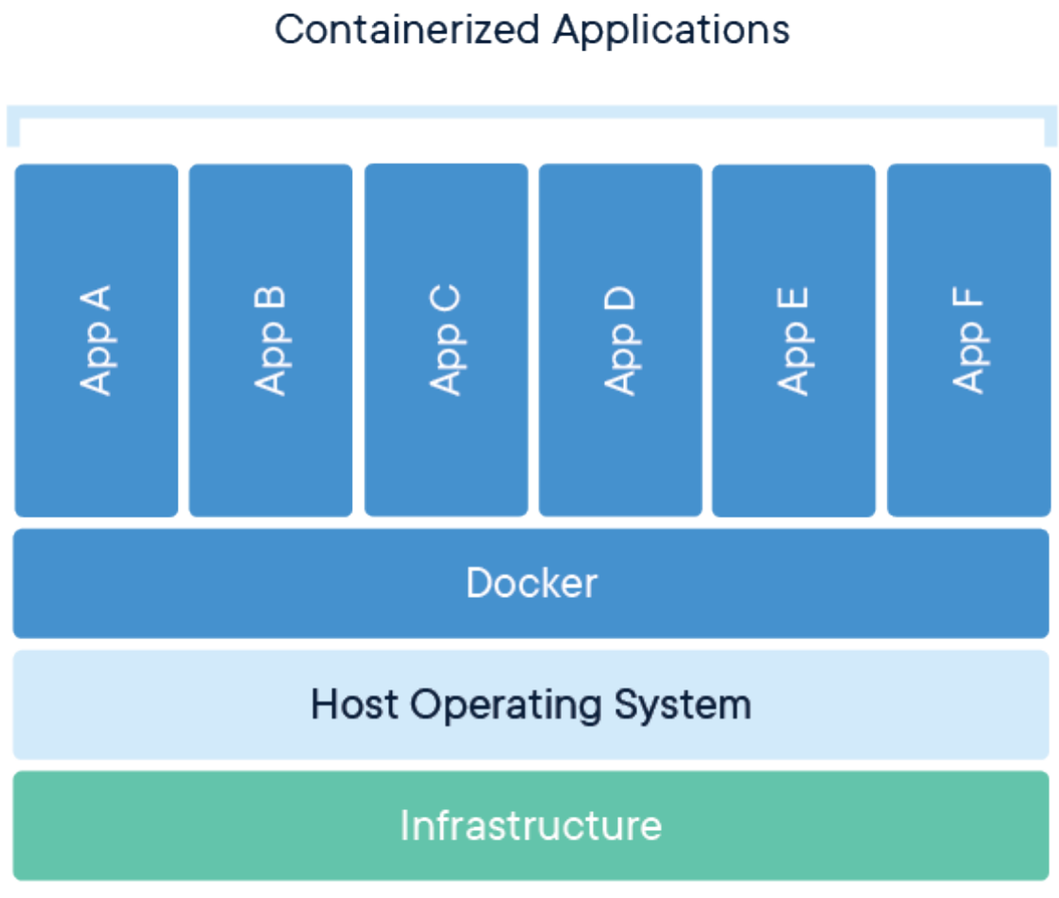
I assist students with personal laptop issues on best-effort basis

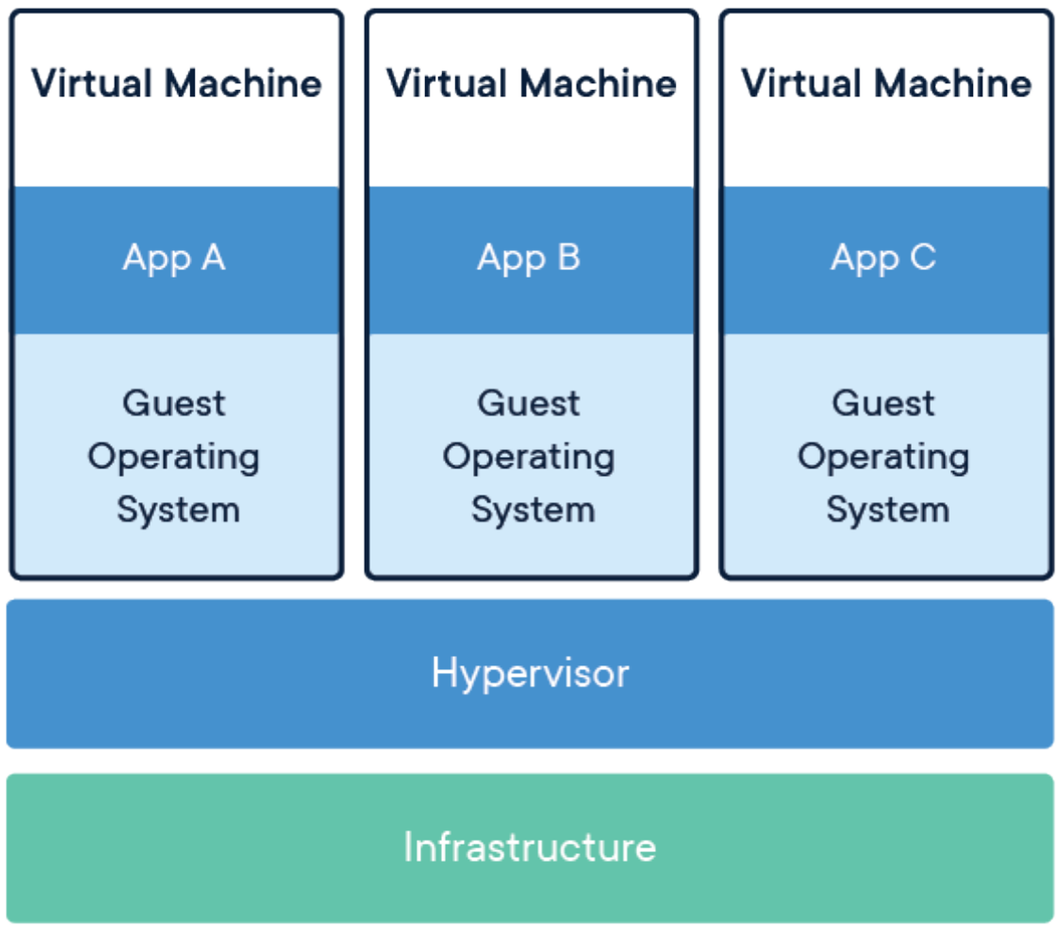
Executing code

- Some people that have tablet two-in-one “laptops” will have trouble with containers
  - This is because the tablet is basically a large phone with a keyboard, and we need a “real” computer capable of running virtual machines
  - For those at UB: If you do not have a powerful enough laptop, you can use the SENS cluster.
  - For those elsewhere: singularity is installed at the LPC so if you have an account there, instructions can be provided by the LPC staff to run the container.

Containers

- Can also make use of containers
  - Virtual “machine” but for applications





Containers

- Why bother?
  - So you don't have to spend time configuring software. It just works.

- Instructions:
  - Get docker : https://docs.docker.com/install/
    - (shortcuts: https://download.docker.com)
  - Follow installation instructions (OS dependent)
  - Once it is running, open your "Terminal" app in your laptop / host machine, and execute these:

```
git clone https://github.com/ubsuny/CompPhys.git
chmod a+w CompPhys
cd CompPhys
./runDocker.sh ubsuny/compphys:latest
```

Containers

- You will get a command line just like a linux computer (running Ubuntu)

```
% ./runDocker.sh srappoccio/compphys:latest
192.168.1.231 being added to access control list
compphys@55d39ad05e02:/Users/rappoccio/dockers/CompPhys$
```

- (Your IP, hash details, and user directory will be different)

UNIX/LINUX

- Operating systems (like Windows or Mac OS)

- We will be using LINUX (an offshoot of UNIX)
  - Invented by Linus Torvalds in 1991
  - Mac OS X is built upon LINUX
- EdX course from the creator here :
  - https://www.edx.org/course/introduction-linux-linuxfoundationx-lfs101x-0#!
- Linux is completely open source : you can modify it at your will and debuggers are also free

- Predominantly command line tools

- Cheat sheet on UBLearns (or indico for FNAL people)

Linux Shells

- We will be using an interface to Linux called a “shell”
- It is a command-line interpreter : you type, it executes

- Two major options are “bash” (as in, smash) and “csh” (like “sea shell”, modern version is “tcsh”, “tea sea shell”)
- For the most part, few differences with respect to this class, you can use either one.

- Major difference : environment variables syntax
  - bash: export X=value
  - tcsh: setenv X value
- That’s about it for the purposes of this class

Jupyter

- We will (judiciously) use Jupyter notebooks
- Also can be installed on your own machines if you want
- Nice little package for code and documentation at once
  - Note: This is usually not for “industrial strength” software, but for prototyping and communication!

Jupyter

- We will (judiciously) use Jupyter notebooks
- Also can be installed on your own machines if you want
- Nice little package for code and documentation at once
  - Note: This is usually not for “industrial strength” software, but for prototyping and communication!
- To run: `jupyter notebook --ip 0.0.0.0 --no-browser`

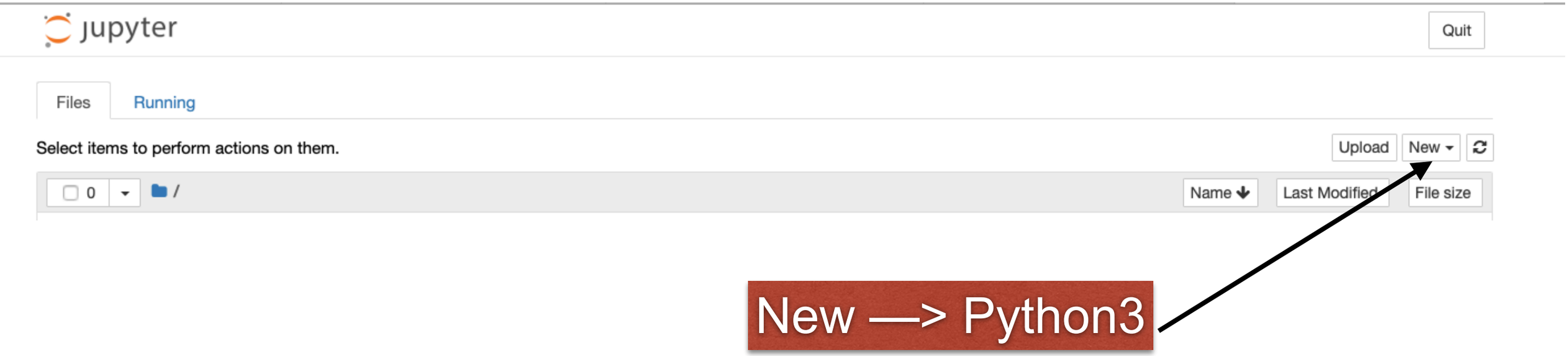
```
compphys@e0e70fd1831c:/Users/rappoccio/dockers/CompPhys$ jupyter notebook --ip 0.0.0.0 --no-browser
[I 19:59:18.332 NotebookApp] Writing notebook server cookie secret to /results/.local/share/jupyter/runtime/notebook_cookie_secret
[I 19:59:18.499 NotebookApp] Serving notebooks from local directory: /Users/rappoccio/dockers/CompPhys
[I 19:59:18.499 NotebookApp] Jupyter Notebook 6.4.12 is running at:
[I 19:59:18.499 NotebookApp] http://e0e70fd1831c:8888/?token=a31edcbb72eb3b6f3678471fd6ed9fbe98a7b7e419193cb1
[I 19:59:18.499 NotebookApp]  or http://127.0.0.1:8888/?token=a31edcbb72eb3b6f3678471fd6ed9fbe98a7b7e419193cb1
[I 19:59:18.499 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 19:59:18.502 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///results/.local/share/jupyter/runtime/nbserver-10-open.html
    Or copy and paste one of these URLs:
        http://e0e70fd1831c:8888/?token=a31edcbb72eb3b6f3678471fd6ed9fbe98a7b7e419193cb1
     or http://127.0.0.1:8888/?token=a31edcbb72eb3b6f3678471fd6ed9fbe98a7b7e419193cb1
[I 19:59:26.391 NotebookApp] 302 GET /?token=a31edcbb72eb3b6f3678471fd6ed9fbe98a7b7e419193cb1 (172.17.0.1) 0.500000ms
```

Copy and paste into web browser

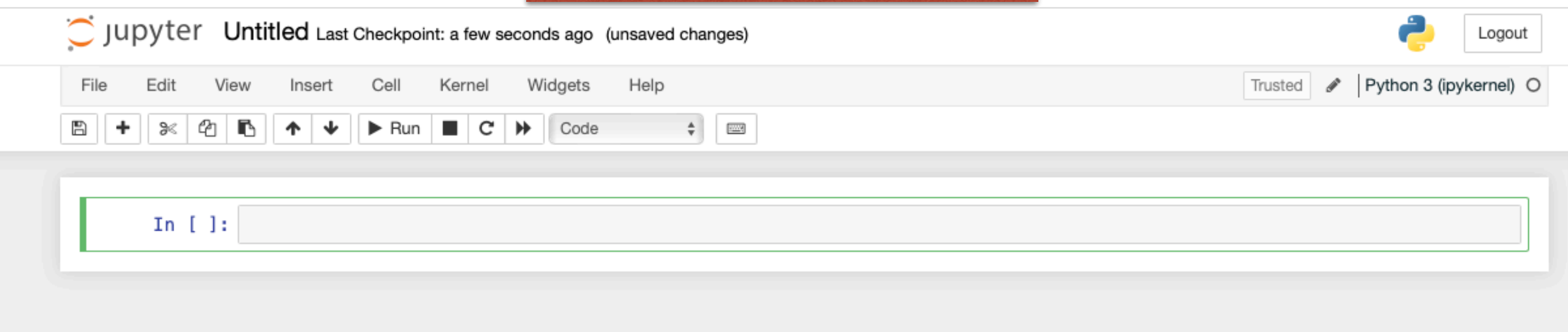
Jupyter

- Now you're running jupyter locally:

Jupyter



Gives you a notebook:



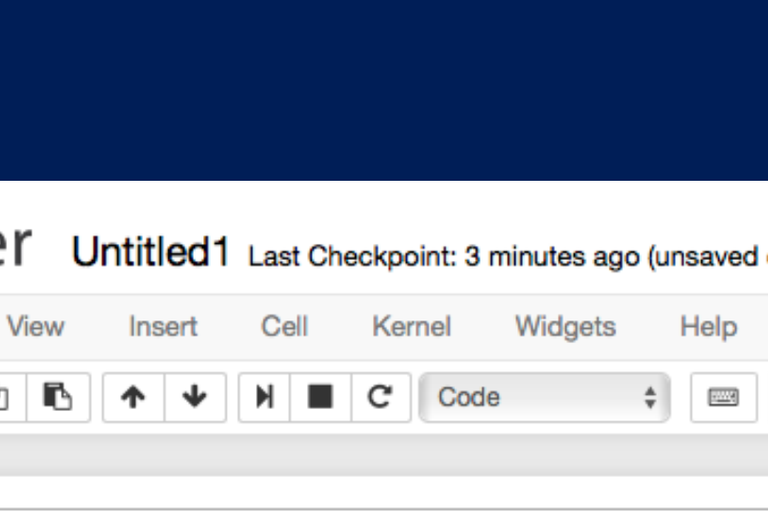
Jupyter

jupyter Untitled1 Last Checkpoint: 3 minutes ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Python2

Code CellToolbar Dashboard View:

```
In [3]: import matplotlib
        import matplotlib.pyplot as plt
        import array

        x = array.array('f', [1,2,3])
        y = array.array('f', [1,4,9])

        plt.plot(x,y)

        plt.show()
```



Type python

Shift+Enter: It executes

In [ ]:

Copy-and-paste-able:

```
import matplotlib.pyplot as plt
import array
x = array.array('f', [1,2,3])
y = array.array('f', [1,4,9])
plt.plot(x,y)
plt.show
```

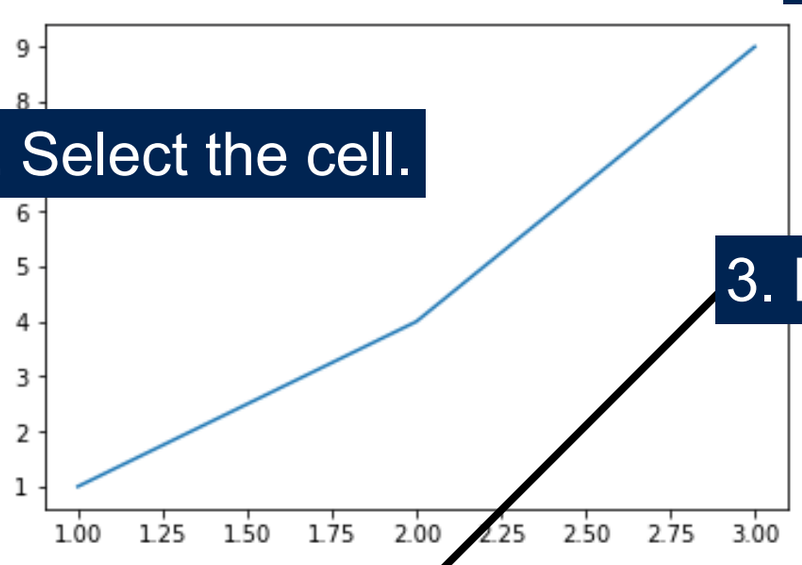
Jupyter

jupyter Untitled1 Last Checkpoint: 37 minutes ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Python2

Markdown CellToolbar Dashboard View:

In [3]:
```
import matplotlib
import matplotlib.pyplot as plt
import array

x = array.array('f', [1,2,3])
y = array.array('f', [1,4,9])

plt.plot(x,y)

plt.show()
```



2. Change cell to “Markdown”

1. Select the cell.

3. LaTeX goes in “$(stuff here)$”

```
This is a markdown cell.

You can use $\LaTeX$ like this : $y = x^2$. Note of caution: it uses Mathjax, so not
everything is supported.
```

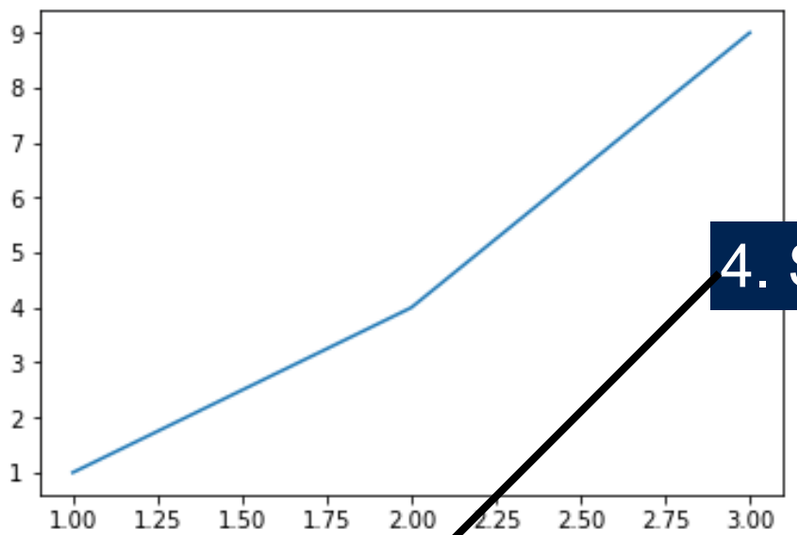
Jupyter

jupyter Untitled1 Last Checkpoint: 37 minutes ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Python2

Markdown CellToolbar Dashboard View:

In [3]:
```
import matplotlib
import matplotlib.pyplot as plt
import array

x = array.array('f', [1,2,3])
y = array.array('f', [1,4,9])

plt.plot(x,y)

plt.show()
```



4. Shift+Click, and you get the markdown

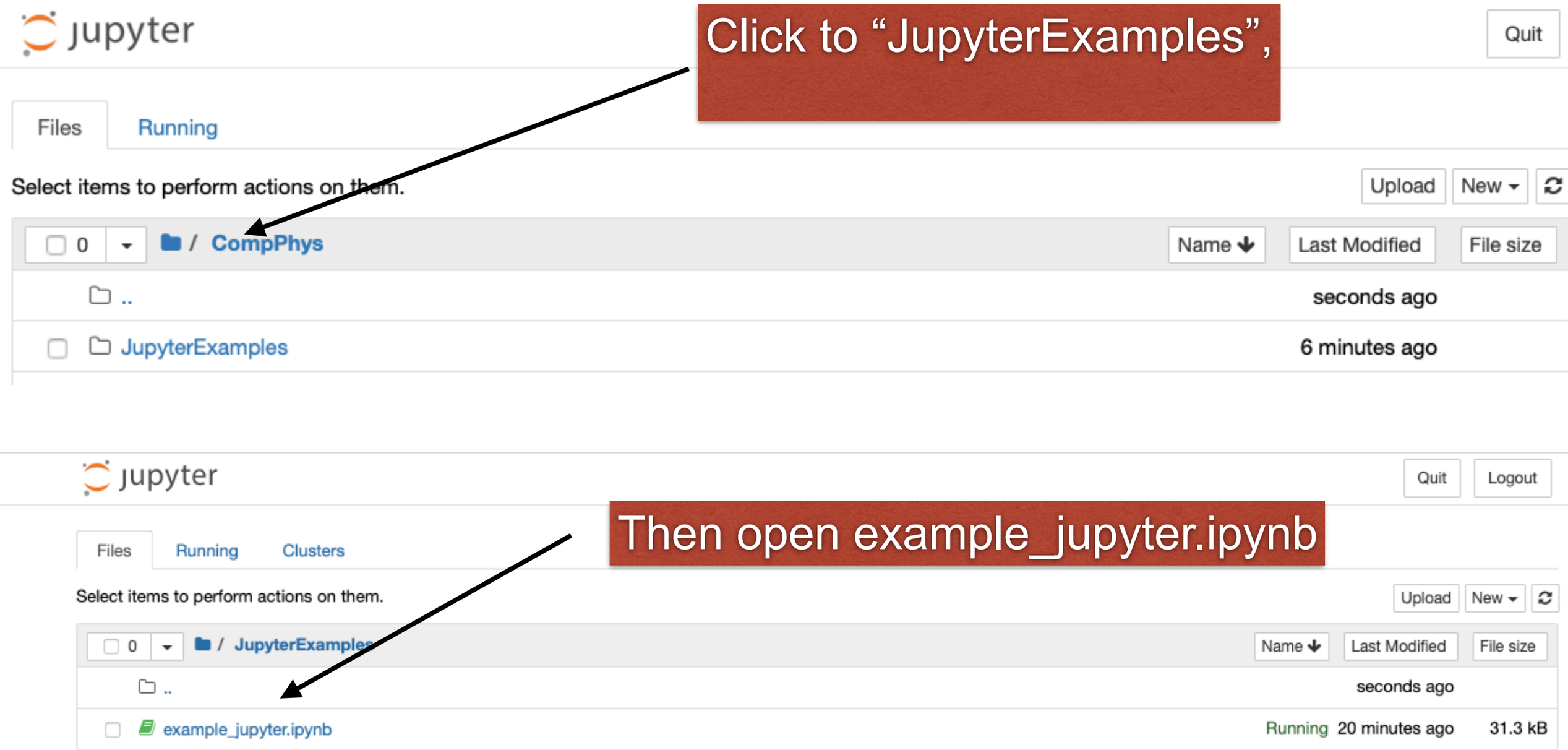
This is a markdown cell.

You can use $\LaTeX$ like this : $y = x^2$. Note of caution: it uses Mathjax, so not everything is supported.

Jupyter

- But wait, it gets better! If you want, you can use the Jupyter terminal directly instead of the linux shell.

Jupyter


jupyter
Quit
Click to “JupyterExamples”,
Files
Running
Select items to perform actions on them.
Upload
New
0
/ CompPhys
Name
Last Modified
File size
..
seconds ago
JupyterExamples
6 minutes ago
jupyter
Quit
Logout
Then open example_jupyter.ipynb
Files
Running
Clusters
Select items to perform actions on them.
Upload
New
0
/ JupyterExamples
Name
Last Modified
File size
..
seconds ago
example_jupyter.ipynb
Running
20 minutes ago
31.3 kB

Jupyter

VIDIA jupyter example_jupyter (autosaved) Terminate Session

File Edit View Insert Cell Kernel Widgets Help Trusted Python3

Run Code Dashboard View:

Example using Jupyter

We will generate linear data with Gaussian noise and fit it to a straight line.

First, we import the libraries.

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

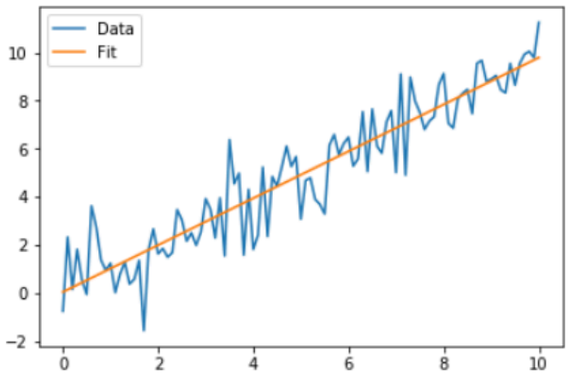
Next, we create some simple x and y data from a linear model with some small Gaussian noise.

In [2]:
```
N=100
x = np.linspace(0,10,101)
y = np.random.normal(scale=1,loc=x)

p, residuals, _, _, _ = np.polyfit(x, y, 1, full=True)
chisq_dof = residuals / (len(x) - 2)

print ('Fit results: y = %3.2f x + %3.2f' % (p[0], p[1]) )
```

```
Fit results: y = 0.98 x + 0.04
```

Finally, we plot the results.

In [3]:
```
yfit =  p[0] * x + p[1]
plt.plot(x,y, label='Data')
plt.plot(x,yfit, label='Fit')
plt.legend()
plt.show()
```



And ta-da! We did a jupyter.

To run:

Kernel:
Restart and Run All

Boom! Jupyter-ed.

Linux Command Line

- Basics : Listing directory contents : “ls”, like “list”

```
$ ls
$
```

  - Empty right now

- Basics : Echoing content : “echo” :

```
$ echo "Why am I here?"
Why am I here?
```

- Basics : Saving what you type: the “pipe” command : “>”

```
$ echo "Who am I?" > stuff.txt
$ echo “Why am I here?” > stuffagain.txt
```

- No output… what happened? “ls” again:

```
$ ls
stuff.txt stuffagain.txt
```

- Aha, there it is. What’s in it?

Linux Command Line

- Reading files : “more”, “less”, and “cat”:
  - “cat” : concatenates files together (if just one, reads it)

```
$ cat stuff.txt
Who am I?
```

  - “less”: reads a file and paginates it

```
$ less stuff.txt
Who am I?
~
```

  - “more” : reads multiple files separately

```
$ more stuff.txt
Who am I?
```

Linux Command Line

– “cat” will concatenate:

```
$ cat stuff.txt stuffagain.txt
Who am I?
Why am I here?
```

– But “more” does more:

```
$ more stuff.txt stuffagain.txt
::::::::::::::
stuff.txt
::::::::::::::
Who am I?
::::::::::::::
stuffagain.txt
::::::::::::::
Why am I here?
$
```

Linux Command Line

- What if you want to copy or move files?
- Copy : “cp” :

```
$ cp stuff.txt stuff1.txt
$ ls
stuff1.txt  stuffagain.txt  stuff.txt
$ cat stuff1.txt
Who am I?
```

  - File “stuff1.txt” is a copy of “stuff.txt”
  - They are duplicates. “stuff.txt” is retained.

- Move : “mv” :

```
$ ls
stuff2.txt  stuffagain.txt  stuff.txt
$ mv stuff1.txt stuff2.txt
$ cat stuff2.txt
Who am I?
```

  - File “stuff2.txt” is there, but “stuff1.txt” is gone
  - Also known as “renaming”!

Linux Command Line

- Removing files : “rm”

```
$ rm stuff2.txt
rm: remove regular file `stuff2.txt'? y
$ ls
stuffagain.txt  stuff.txt
```

  - Gone forever. Really, really, really gone. No “trash bin”. No “restore”. No “but professor, can’t I undo that?” No. You cannot. Bits are gone.

- Backups in LINUX are VERY IMPORTANT TO HAVE

  - Example:

```
$ cp stuff.txt stuff_backup_27jan2017.txt
```

  - OR! Use version control (more on that later)

Linux Command Line

- What if I hate typing files one at a time?
  - Then Linux is the best operating system for you

- Huge number of shortcuts in LINUX :
  - tab completion : If you start typing and press “TAB”, it will list the possible completions, or if there is only one, complete it for you
  - “up” key : reproduces previous line in Terminal
  - aliases : can define your own shortcuts

  - wildcards!

Hands on!

- Open terminal, go to “CompPhys/LinuxOverview”
- Look at “commandline1.sh” file:

```
% more commandline1.sh
echo "Always look on the bright side of life" > bright.txt
echo "If life is jolly rotten, there's something you've
forgotten" > forgotten.txt
cat bright.txt forgotten.txt > song.txt
cat song.txt=
```

- Execute:

```
% bash commandline1.sh
```

- Output:

```
Always look on the bright side of life
If life is jolly rotten, there's something you've forgotten
```

- Type “ls”:

```
bright.txt commandline1.sh forgotten.txt song.txt
```

- Demonstrates: more, ls, and cat commands, files, and piping

Linux Command Line

- Wildcards : http://www.tldp.org/LDP/GNU-Linux-Tools-Summary/html/x11655.htm

– ? : single character

```
$ ls ?right.txt
bright.txt
```

– * : any number of characters

```
$ ls *.txt
bright.txt  forgotten.txt  song.txt  stuffagain.txt
stuff_backup_27jan2017.txt  stuff.txt
```

```
$ ls s*.txt
song.txt  stuffagain.txt  stuff_backup_27jan2017.txt  stuff.txt
```

Linux Command Line

- Wildcards:

– [] : specifies a range

```
$ echo "song1" > song1.txt
$ echo "song2" > song2.txt
$ echo "song3" > song3.txt
$ cat song[1-2].txt
song1
song2
```

– {} : name or wildcard

```
$ echo "song4" > whoops.ugh
$ cat {song*.txt,whoops.ugh}
song1
song2
song3
Always look on the bright side of life
If life is jolly rotten, there's something you've forgotten
song4
```

Linux Command Line

- Command line options :
  - What if you like “rm”, but hate to confirm each removal?
  - “rm” has OPTIONS you can add
- Options usually come in two forms :
  - Single dash, single letter : “ls -r”
  - Two dashes, many letters: “ls —help”
  - Most commands come with “—help”, so you can read the options you have

Linux Command Line

- Example : “ls”:

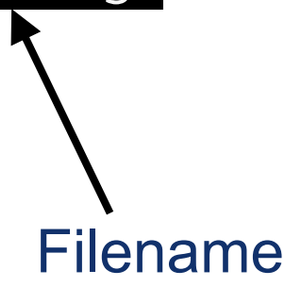
```
$ ls --help
Usage: ls [OPTION]... [FILE]...
List information about the FILEs (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort.

Mandatory arguments to long options are mandatory for short options too.
  -a, --all                  do not ignore entries starting with .
  -A, --almost-all           do not list implied . and ..
      --author               with -l, print the author of each file
  -b, --escape               print octal escapes for nongraphic characters
      --block-size=SIZE      use SIZE-byte blocks.  See SIZE format below
```

- Common options
  - “-l” : long listing
  - “-a” : list all
  - “-t” : list by time
  - “-r” reverse

Linux Security

- What the heck is all this stuff?

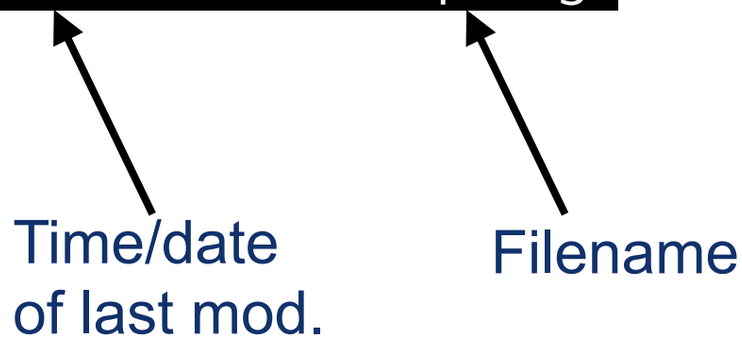
```
-rw-------  1 srrappoc phyfac    6 Jan 27 11:14 whoops.ugh
```



Linux Security

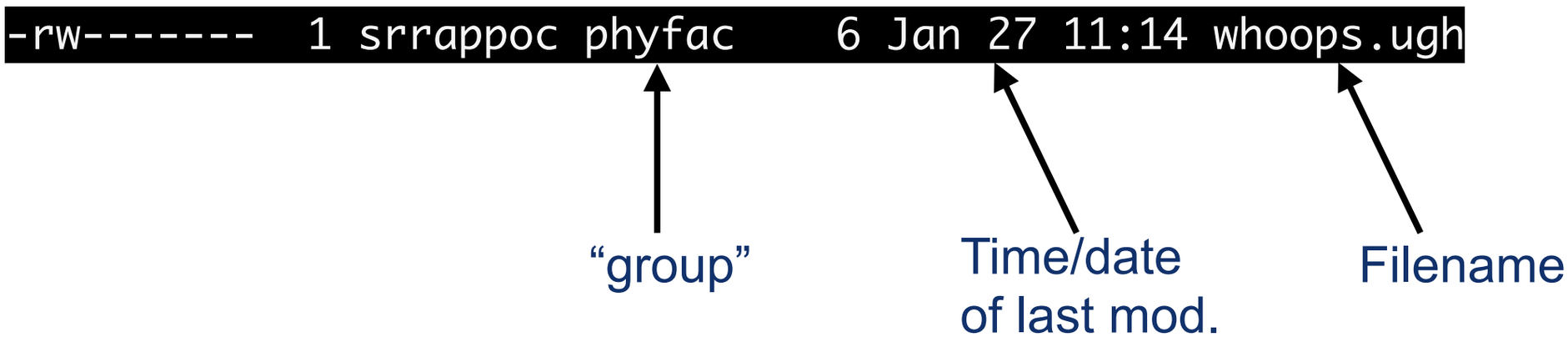
- What the heck is all this stuff?

```
-rw-------  1 srrappoc phyfac    6 Jan 27 11:14 whoops.ugh
```



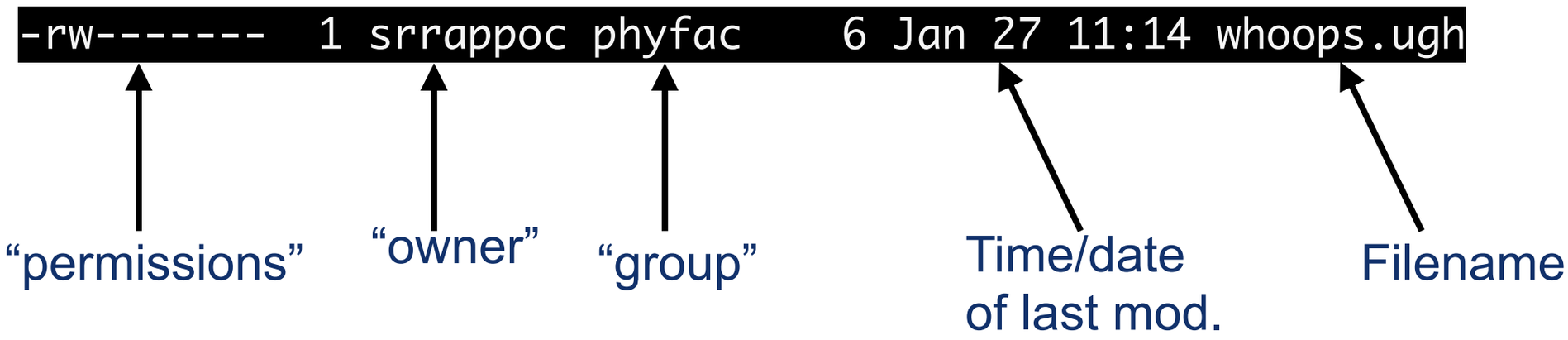
Linux Security

- What the heck is all this stuff?



Linux Security

- What the heck is all this stuff?



Linux Security

- Levels of security :
  - All : everyone
  - Group : a group of users
  - Owner : the owner of the file

- Access levels :
  - Read : Can read from file
  - Write : Can write to file
  - Execute : Can execute if it is a program

- Can set different access levels for different levels
  - Most common example: “Owner” can read+write, “Group” can read, “All” can do nothing

Linux Security

- You can change the permissions, the owners, or the groups of files:

  - chown (change owner or group) :
    - usage : chown user:group file
    - change owner of file to “user” and group to “group”
  - chmod (change permission)
    - usage examples :
      - chmod a+x file (give execute permission to “all”)
      - chmod g+w file (give write permission to “group”)
      - chmod a+wrx file (give read/write/execute permission to anyone)

Linux Directories

- To make directories, use “mkdir”:

```
$ mkdir testdir
$ ls
bright.txt  commandline1.sh   forgotten.txt   song.txt    testdir
```

  - To remove them, “rmdir” (must be empty)

- You can move files from one directory to another with “mv” just like before:

```
$ mv song.txt testdir/
```

- And you can use “ls” to look at the directory:

```
$ ls testdir/
song.txt
```

Linux Directories

- The directory you are “in” is called the “current directory”
- To change your current directory, use “cd”:

```
$ cd testdir
/nsm/home/srrappoc/testdir
$ ls
song.txt
```

- To go to the directory “up” the chain, do “cd ..”:

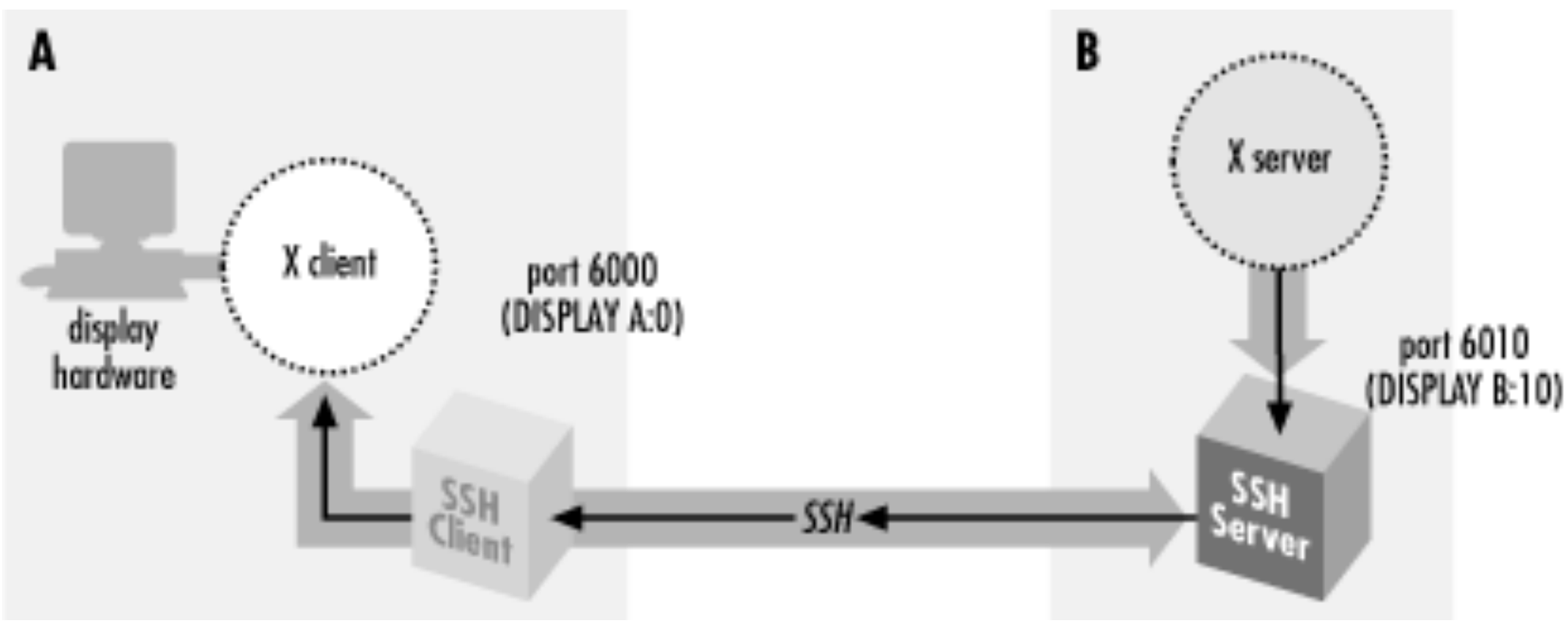
```
$ cd ..
```

- There are shortcuts for “cd” :
  - “cd -“ go to previous directory
  - “cd” (no arguments) go to home directory

- You can also print the working directory (pwd)

Editors

- AKA : why is it so f*ing hard to type stuff in Linux?
  - It isn't, you're just not thinking about it right :)
  - If you're sitting at the same terminal you're executing commands on, your server is the same as your client
  - If you're using ssh, you are trying to send a ton of graphical information over the net to type



Editors

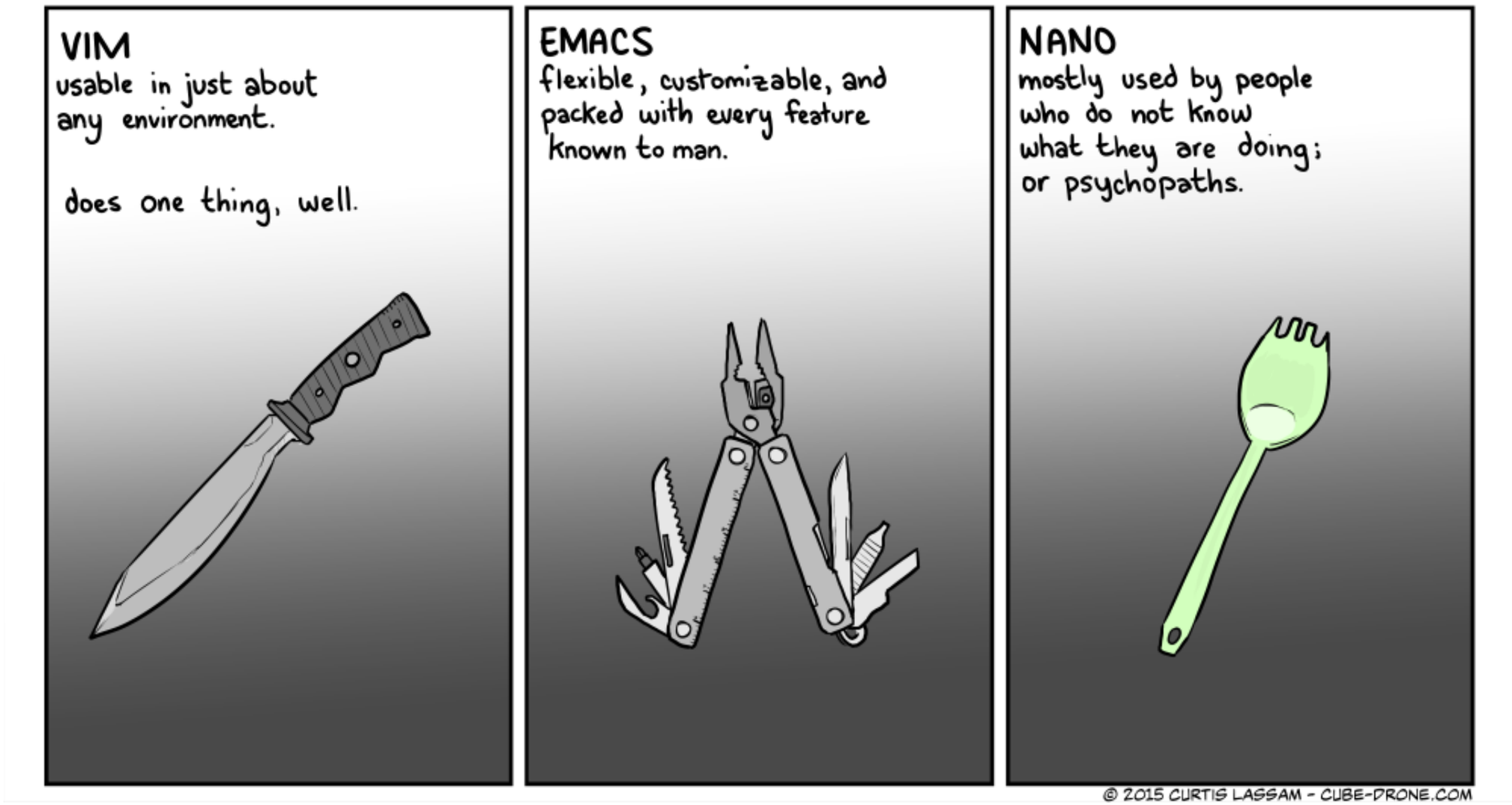
- Linux comes with the “X11” graphics package : that’s how everything works
  - This is also the basis for Mac OS’s GUI

- Many editors are available, and many have “inline text” and “X11” modes

- Now we come to one of the greatest scientific controversies of all time:

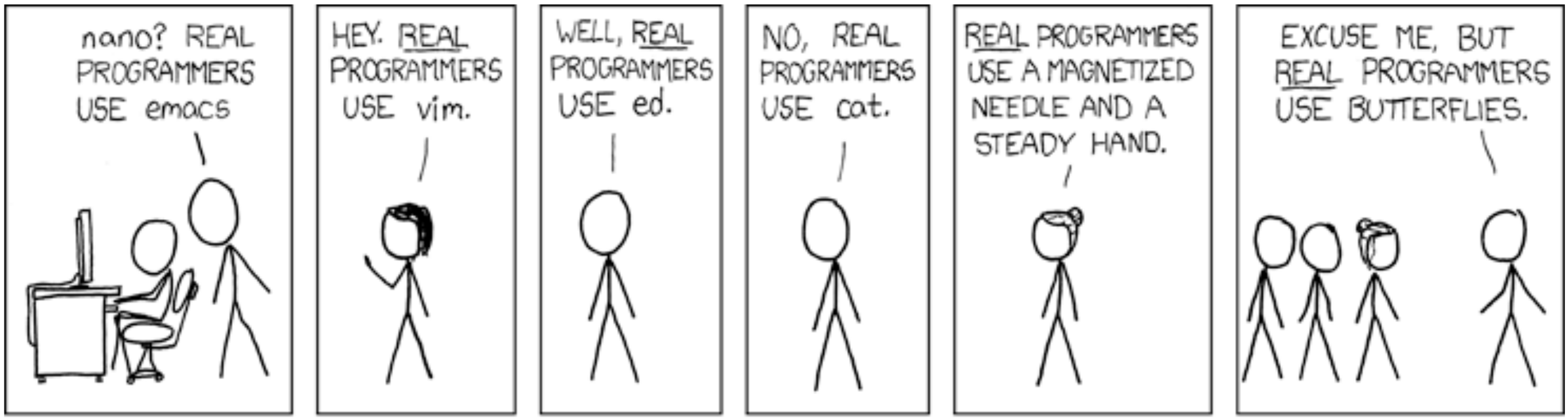
THE EDITOR WARS

Editors



http://cube-drone.com/comics/c/holy-war

Editors


nano? REAL PROGRAMMERS USE emacs
HEY. REAL PROGRAMMERS USE vim.
WELL, REAL PROGRAMMERS USE ed.
NO, REAL PROGRAMMERS USE cat.
REAL PROGRAMMERS USE A MAGNETIZED NEEDLE AND A STEADY HAND.
EXCUSE ME, BUT REAL PROGRAMMERS USE BUTTERFLIES.
THEY OPEN THEIR HANDS AND LET THE DELICATE WINGS FLAP ONCE.
THE DISTURBANCE RIPPLES OUTWARD, CHANGING THE FLOW OF THE EDDY CURRENTS IN THE UPPER ATMOSPHERE.
THESE CAUSE MOMENTARY POCKETS OF HIGHER-PRESSURE AIR TO FORM,
WHICH ACT AS LENSES THAT DEFLECT INCOMING COSMIC RAYS, FOCUSING THEM TO STRIKE THE DRIVE PLATTER AND FLIP THE DESIRED BIT.
NICE.
'COURSE, THERE'S AN EMACS COMMAND TO DO THAT.
OH YEAH! GOOD OL' C-x M-c M-butterfly...
DAMMIT, EMACS.

Editors

- I am personally firmly in the “Emacs” zone of the world
- If you are a vi or ed user, you can leave now
  - Kidding!!! *coughsortofcough*

- I can teach you the basics of emacs, and you should learn the rest on your own. You’re scientists, that’s how it goes. It’s like learning to walk, or ride a bike, or hazing or something.

Editors

- emacs :
  - X mode : emacs filename.txt &
    - Edit away as you want.
  - Terminal mode : emacs -nw filename.txt
    - “-nw” is “no window”
  - Tutorial : https://www.gnu.org/software/emacs/tour/
- vi / vim:
  - vi filename.txt
  - I suck at vi, so just go here:
    - https://scotch.io/tutorials/getting-started-with-vim-an-interactive-guide

Editors

- If you're terrified of learning how to use these, just use "nano". The interface is straightforward, but has extremely few features.

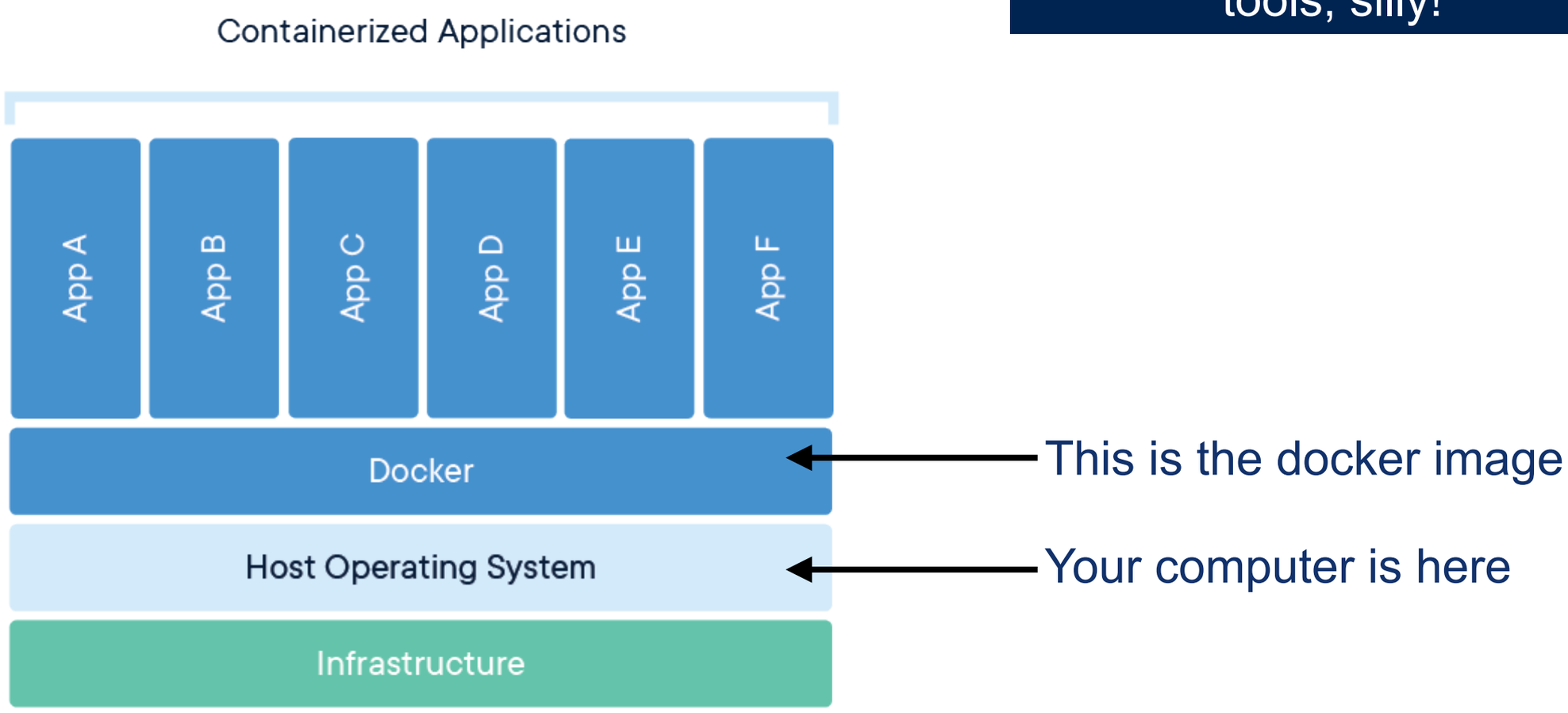
- Some of you young whipper snappers will want to use a more modern editor like sublime or xcode
  - If you know out how to use them, go for it, but I won't help you! :)

Editing: Docker

- There is no emacs in our docker image
- There is no vi in our docker image
- There is no editor at all

- WTH Sal?!!?

Use the host operating system's editing tools, silly!



Editing: Docker

- Open the files directly in your host operating system (i.e. the normal way you edit stuff on your computer)

```
compphys@d4fa2e3932fb:/Users/rappoccio/dockers/CompPhys$
```

The file is here on your host operating system, just edit in a different terminal!