
Python Intro



Bob Baxley, PhD & Vince Emanuele, PhD



Slides at <https://goo.gl/a8oMiu>

Please install Google Chrome if you don't already have it!



How Anidata came to be



Dark Data and Improving Human Rights in Fulton County

Bob Baxley, Ph.D.
Anidata
bob.baxley@gmail.com
December 2016

There are many people with both the Aptitude and Motivation to become data scientists

They are missing:

- 1) good data science mentors
- 2) exposure and experience working on real projects

We started Anidata as a 501c3 Nonprofit started to connect Professional and Aspiring Data Scientists with problems involving Social Good.

<http://anidata.org/>

About the Instructors



Bob Baxley
Chief Engineer,
Bastille

Github: **gte620v**
Twitter: **@bob_baxley**
Email: **bob.baxley@gmail.com**
Web: **rjbaxley.com**

Bob is the Chief Engineer at Bastille where he helps build systems to sift through massive amounts of radio frequency data. He joined Bastille in 2014 shortly after the company was founded and leads the Data Science and Signal Processing teams.

Bob has more than 10 years of experience implementing machine learning systems with an emphasis on cognitive radios. Prior to joining Bastille, Bob was the Director of the Software Defined Radio Lab at the Georgia Tech Research Institute. In that role, Bob led GTRI's team to second place out of 90 international competitors in the DARPA Spectrum Challenge.

In 2008 Bob earned his PhD degree in Electrical Engineering from Georgia Tech. During his graduate work, he was recognized with the Sigma Xi Best Thesis award, the CSIP PhD Research Award, and the NSF GRFP Award. He has co-authored over 70 peer-reviewed papers, is the inventor of 17 patents, and formerly served as an Associate Editor for Digital Signal Processing.

In addition to his role at Bastille, Bob is an Adjunct Faculty member in the School of ECE at Georgia Tech. He also volunteers as a Board Member of the data science non-profit organization, Anidata.



Vincent Emanuele II, PhD
Manager of Data Sciences,
Wellcentive

Github: **vae2**
Twitter: **@VinceEmanuele**
vincent.emanuele@gmail.com

Vincent Emanuele received a PhD in Electrical and Computer Engineering from Georgia Tech. His research involved the development and application of novel data science algorithms to a variety of areas in medicine and biology, including the analysis of genomic data, proteomic data, ECG data, electrophysiological data, and the application of wavelets and neural networks to interpret data in highthroughput mass spectrometry experiments studying cervical cancer. This work led to several publications in refereed journals and a full patent application (in progress). After 6.5 years as a visiting researcher at the Centers for Disease Control and Prevention, he joined the team at Wellcentive, a leading Population Health Management company, in January 2013.

As Lead Data Quality Architect in 2013, Vincent developed the company strategy for data quality, researched entity resolution strategies, started a data quality/data management team, and designed and launched a data quality services offering. In the 19 months from Oct 2013 to Apr 2015, the Data Quality Services offering grew from 1 to 24 customers and surged well past the \$1M revenue mark.

Further, Vincent drove the creation of the company's first Data Governance Committee in 2014. The data governance committee is an interdisciplinary group of employees in the company that come together on a regular basis to review clinical mappings between different medical ontologies such as ICD9, SNOMED, CPT, LOINC, and so forth. In February 2015 Vincent became the Manager of Data Sciences, and in June 2014 he began transitioning the DQ Services program to a new leader and moved on to found the Data Science team at Wellcentive.

Vincent founded the data science team at Wellcentive in August 2014. The data science team is the first R&D unit at Wellcentive, and is actively involved in bringing innovative applications to market in the areas of Suggestive Care, Healthcare System Benchmarking, and Natural Language Processing

Dan Robertson, MPH



Dan Robertson is an aspiring Data Scientist working as a Software Engineer at Tripwire. He has a Masters in Public Health from George Mason University with a specialization in Epidemiology and a graduate certificate in Biostatistics. Dan is a real autodidact, spending much of his free time coding in Python, C++, and the emerging high-performance language Rust. Dan has been involved in and contributed to open-source projects including LibreOffice and Servo.

Github: danlrobertson

Email: danlrobertson89@gmail.com

Bryant Menn



Bryant is a data scientist at Philips Wellcentive working on machine learning applications for patient care management. Prior to being a data scientist, he has worked as a data quality analyst and software engineer at Wellcentive developing an internal analytics platform. He has a bachelors in biomedical engineering from Georgia Tech. In his free time, he is usually tinkering, getting his feet wet in IoT or experimenting with big data analytics/ML on his home server.

LinkedIn: <https://www.linkedin.com/in/bryant-menn-55aa7577>

Github: bmenn

Twitter: @bryant_menn

Email: bryant.menn@gmail.com

Your Background?

Intro

- Name
- Goals for class
- What you do?

Objectives for Class

If you're a beginner

- We want this workshop to help you decide if programming is right for you
- We want this workshop to help you decide if you should invest more time in Python
- We want you to have material in your hands that can serve as your roadmap for self-study

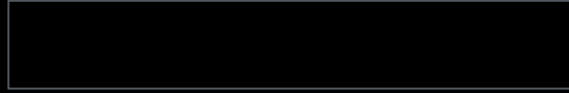
And we want to “teach you how to fish”
(which can be more frustrating at first)



Objectives for Class

If you're not a beginner

- We want this workshop to help you decide if you should invest more time in Python
- We want you to leave with a roadmap for your own self-study to learn how to program in the *python* way
- We want you to see clearly both the strengths and weaknesses of python



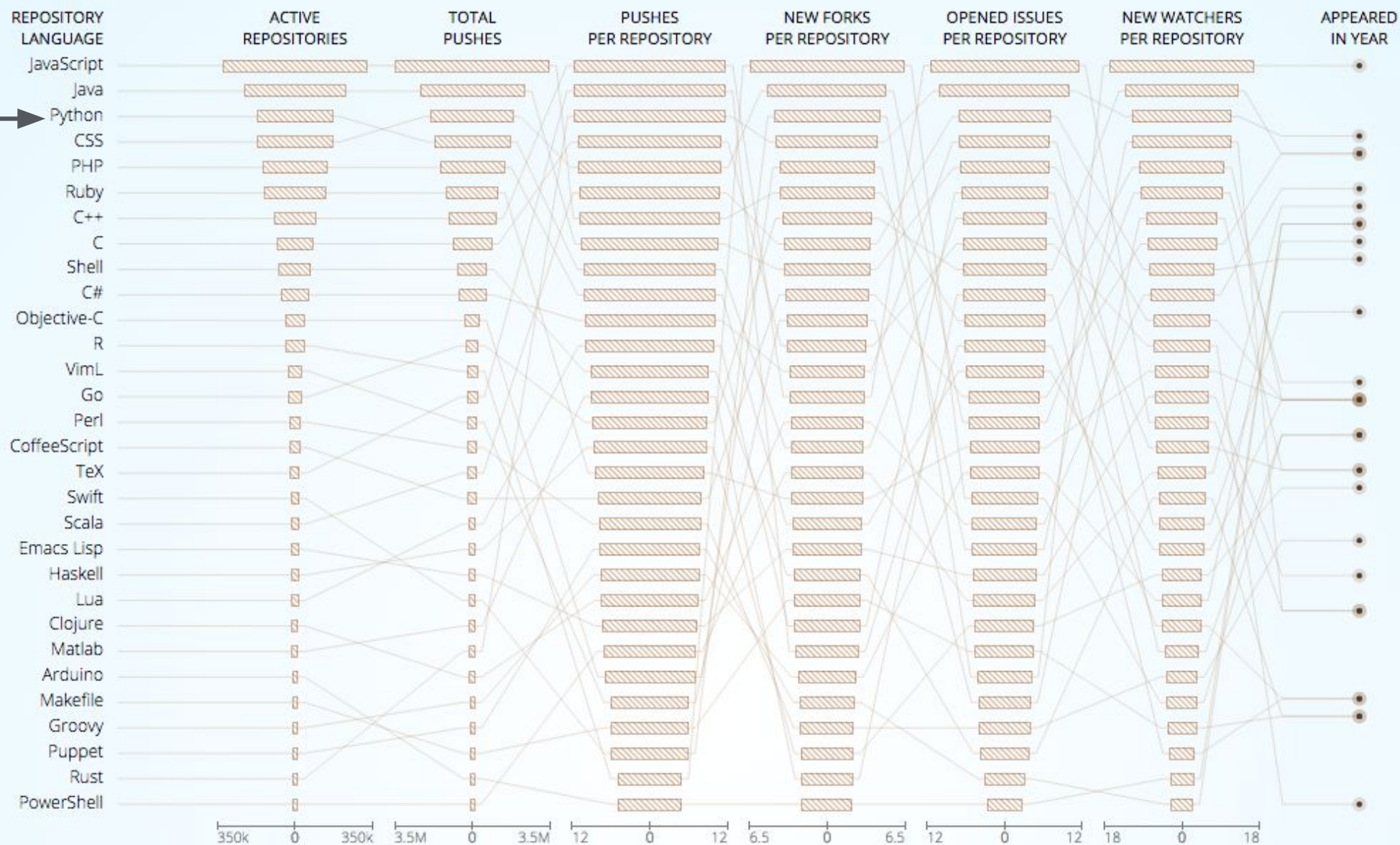
Python?

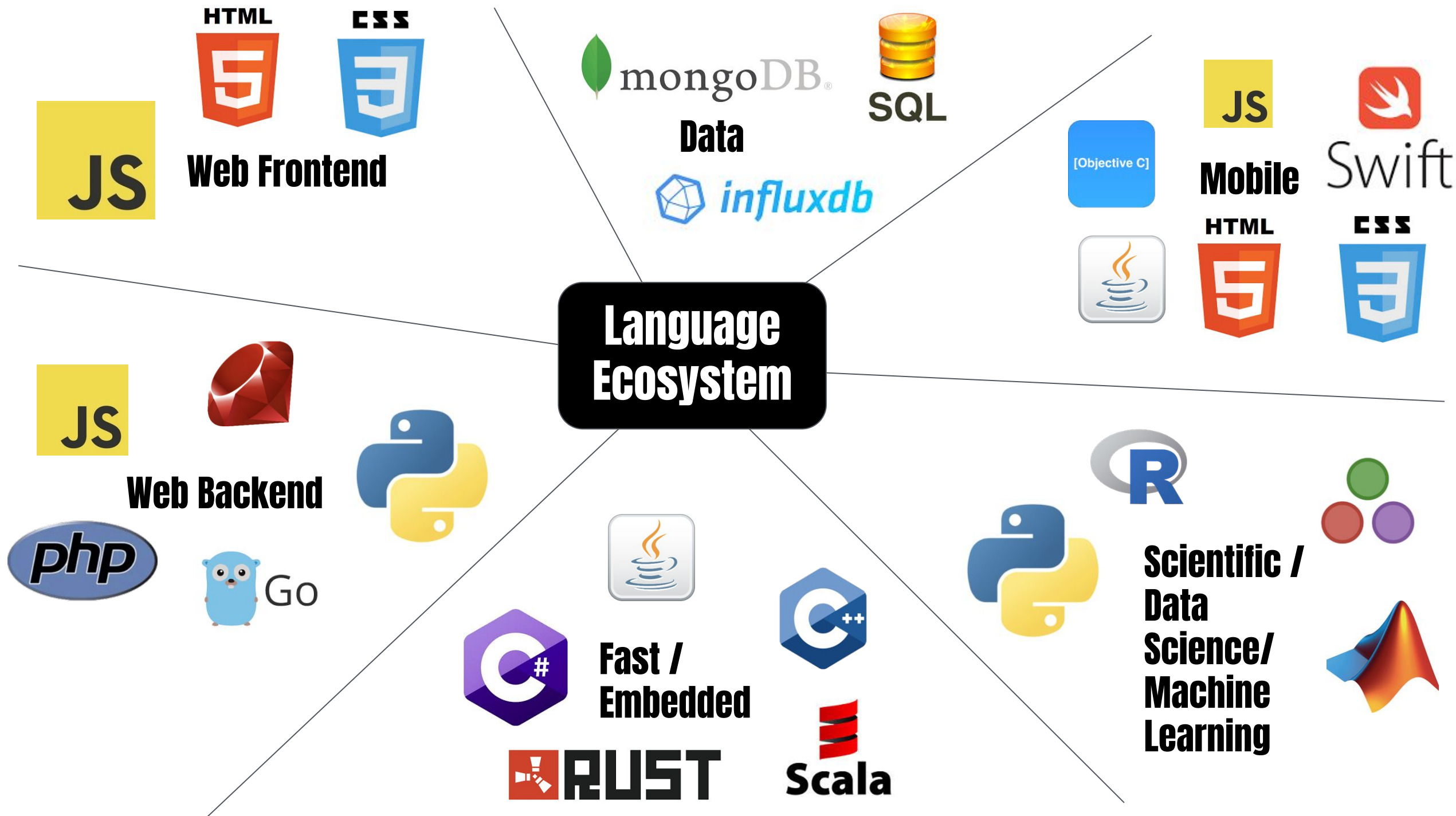


#3



< Q4/14 >





What is Python Good At?

Readability / Easy to Learn & Use

- Python aims to have a simple stripped-down syntax that minimizes boilerplate and other syntactical overhead

Java

```
public class Hello {  
    public static void main(String []args) {  
        System.out.println("Hello World");  
    }  
}
```

See <http://wiki.c2.com/?HelloWorldInManyProgrammingLanguages>

Python

```
print "hello world"
```

Libraries & Examples

- Python has been around long enough and is used in just a variety of applications that there is likely already code or a library built for any application you can imagine.

What is Python Good At? (2)

General Purpose Language

- Python is used for all kinds of programming so it is easy to implement end-to-end systems in python.
- This is in contrast to other languages like Matlab or R, that only do scientific computing well.



What is Python Less Good At?

Speed (...Maybe)

- Python is an interpreted rather than a compiled language, so it is going to be slower than C++.
- It is slower in terms of raw number crunching, but if you factor in development time and maintainability, python shines and is why Google said early on “Python where we can, C++ where we must.”
- Python can call C/C++ modules, which lets us overcome this shortcoming
- Speed is not that relevant in lots of applications. Especially when it is cheap to buy more compute power.

n-body

source	secs
--------	------

<u>Python 3</u>	787.02
-----------------	--------

<u>C++ g++</u>	9.31
----------------	------

spectral-norm

source	secs
--------	------

<u>Python 3</u>	188.83
-----------------	--------

<u>C++ g++</u>	1.99
----------------	------

mandelbrot

source	secs
--------	------

<u>Python 3</u>	273.43
-----------------	--------

<u>C++ g++</u>	1.73
----------------	------

Tooling for Python

Tooling

The Battle

- › Half the battle with learning a new language is understanding the workflow and resources available. To that end, I am going to spend a bit of time on the tooling.
- › Learning the tooling is a productivity multiplier because it will let you grab OSS code for use and for learning

Tools

- › Cloud 9 IDE
- › Git / GitHub
- › Anaconda
- › Jupyter



Environment

Tools

- Developers and developer tools are made to run from the command line
- It can be a big task to set up your environment; a task full of perils
- To avoid these perils, we will just use a cloud virtual machine (VM) so we know we are all on the same page



Tools

Cloud 9 IDE

Cloud 9 IDE

c9.io



Cloud9 IDE
Your code anywhere, anytime

- IDE = Integrated Development Environment
- Most basic version is a text editor (i.e. notebook)
- Fancier IDEs (Sublime, Rodeo, Visual Studio, Spyder):
 - Syntax checking
 - Code completion
 - Compile pipelines
- C9 does some of this, but more importantly, you also get a (small) VM that you can provision how you like
- Note: you MUST use **Google Chrome** or Firefox
- <https://community.c9.io/t/supported-browsers/1988>

C9 is not that special and there are other options like it. We are mostly using it for the class so we are all on the same page.

Cloud 9 IDE

[Hosted workspace](#) Clone workspace Remote SSH workspace Salesforce Google Cloud Platform












☐ **Private**
This is a workspace for your eyes only

☒ **Public**
This will create a workspace for everybody to see

Clone from Git or Mercurial URL (optional)

e.g. ajaxorg/ace or git@github.com:ajaxorg/ace.git

Choose a template

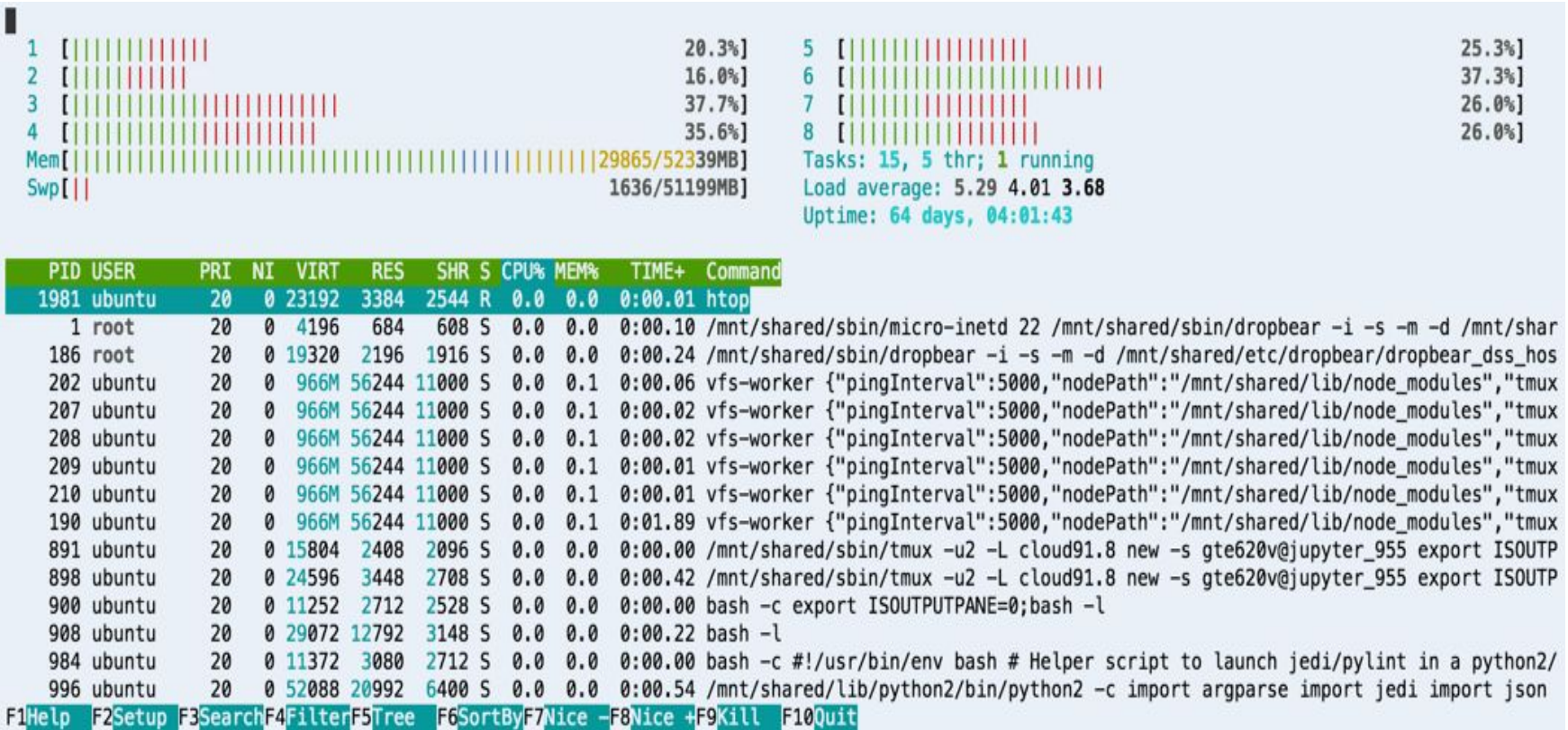
 HTML5	 Node.js	 PHP, Apache &...	 Python	 Django	 Ruby
 C++	 Wordpress	 Rails Tutorial	 Blank	 Harvard Unive...	

Create workspace

Cloud 9 IDE

Bit of Linux

- ▶ run `ls`
- ▶ run `htop`
- ▶ run `df -h`



Cloud 9 IDE

Bit more of Linux

- ▶ run `export NUM=5`
- ▶ run `echo $NUM`
- ▶ run `echo $PATH`

- ▶ run `ssh-keygen`
- ▶ run `cat ~/.ssh/id_rsa.pub`

```
gte620v:~/workspace $ cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCSu6jtgngkNt4ggI7UGetZu2V78+0Iv8v9sJzLhn2cB35i7UYiQrUvjTygMElGcgA0w/AXbXoHli1+88C65Qntns4SxYSQ6jm6topVqhQL
W00ij9nWXrxi7U0nR2wz+CE0FtlipxoV0SNKCvhKeJwvN38Jbi6NYeN+8kkUcLsjNAa+lPcSU06+RfBMro+QyyFtvrGSCkyz1Hc8TPzIO+JSfwLBPlIBrudl4cDrhnFc7ga3r86aukbULPl
0Va7zh80eMy2oDjSfA24lvN665901kQ80+F9vWI/m/6iUF/wA4RU+x+NqEy7mlzyWHrqDSVanMISUttd/DfeREsP0tpP ubuntu@gte620v-jupyter-3368070
```

Cloud 9 IDE

Try Python Shell

- ▶ run `python`
- ▶ run `import math`
- ▶ run `print('Hello World!!')`
- ▶ run `2+2`
- ▶ Press control+d

```
gte620v:~/workspace $ python
Python 3.6.0 |Continuum Analytics, Inc.| (default, Dec 23 2016, 12:22:00)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import math
>>> print('Hello World!!')
Hello World!!
>>> 2+2
4
>>>
gte620v:~/workspace $
```

Cloud 9 IDE

Python File

- Create new file call test.py
 - type `import numpy as np`
 - type `print('Hello World!!')`
 - type `print 2+2`
 - type `print np.arange(10)`
- run `python test.py`

```
gte620v:~/workspace $ python test.py
Hello World!!
4
[0 1 2 3 4 5 6 7 8 9]
gte620v:~/workspace $
```

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: No module named numpy
```

Setup: Anaconda / MiniConda

- Most operating systems ship with python;
- We just saw that you can type `python` in the c9 terminal right now and you will be in a python interpreter.
- **Python does not have a built in package manager for libraries, which are called “packages”.**

So we need to get one:

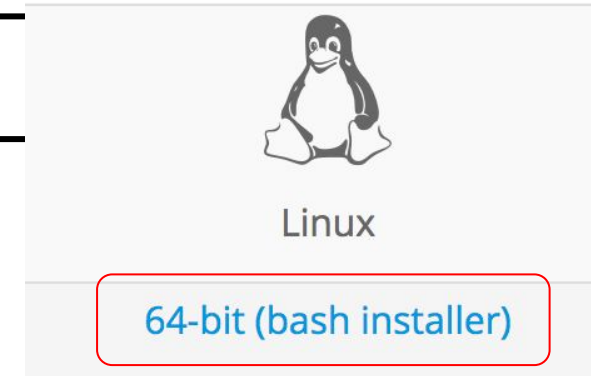
- `pip` is “the manager”
- `conda` is gaining popularity - easier to handle pre-compiled binaries

- We will still use `pip`, but Anaconda will handle installing pip
- “Anaconda” contains the conda manager along with a bunch of popular python modules that are pre-packaged
- Miniconda is Anaconda without the pre-packaged modules



Install: Anaconda / MiniConda

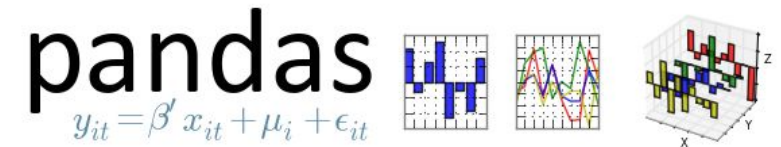
- Go to <https://conda.io/miniconda.html> and right click on
 - “Copy link address”
- Go back to Cloud 9 and copy to terminal
 - `wget https://repo.continuum.io/miniconda/Miniconda3-latest-Linux-x86_64.sh`
- This will copy the miniconda installer to your c9 instance. Now we need to install
 - `bash Miniconda3-latest-Linux-x86_64.sh`
 - Agree to license terms, keep default path, type “yes” for adding to bashrc
- Check
 - Open a new terminal window
 - Type `python`



```
gte620v:~/workspace $ python
Python 3.6.0 |Continuum Analytics, Inc.| (default, Dec 23 2016, 12:22:00)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> █
```

Install: Packages

- Let's install numpy and rerun the numpy code
 - Type `conda install numpy` in the terminal
 - `python test.py`
- Install more packages
 - `conda install pandas matplotlib scikit-learn`
- To check what is installed type
 - `conda list`
- If you get a “No module” error, try a `conda install`
- For things not on conda, you might have to run `pip install`



Tools

Git / GitHub

Git

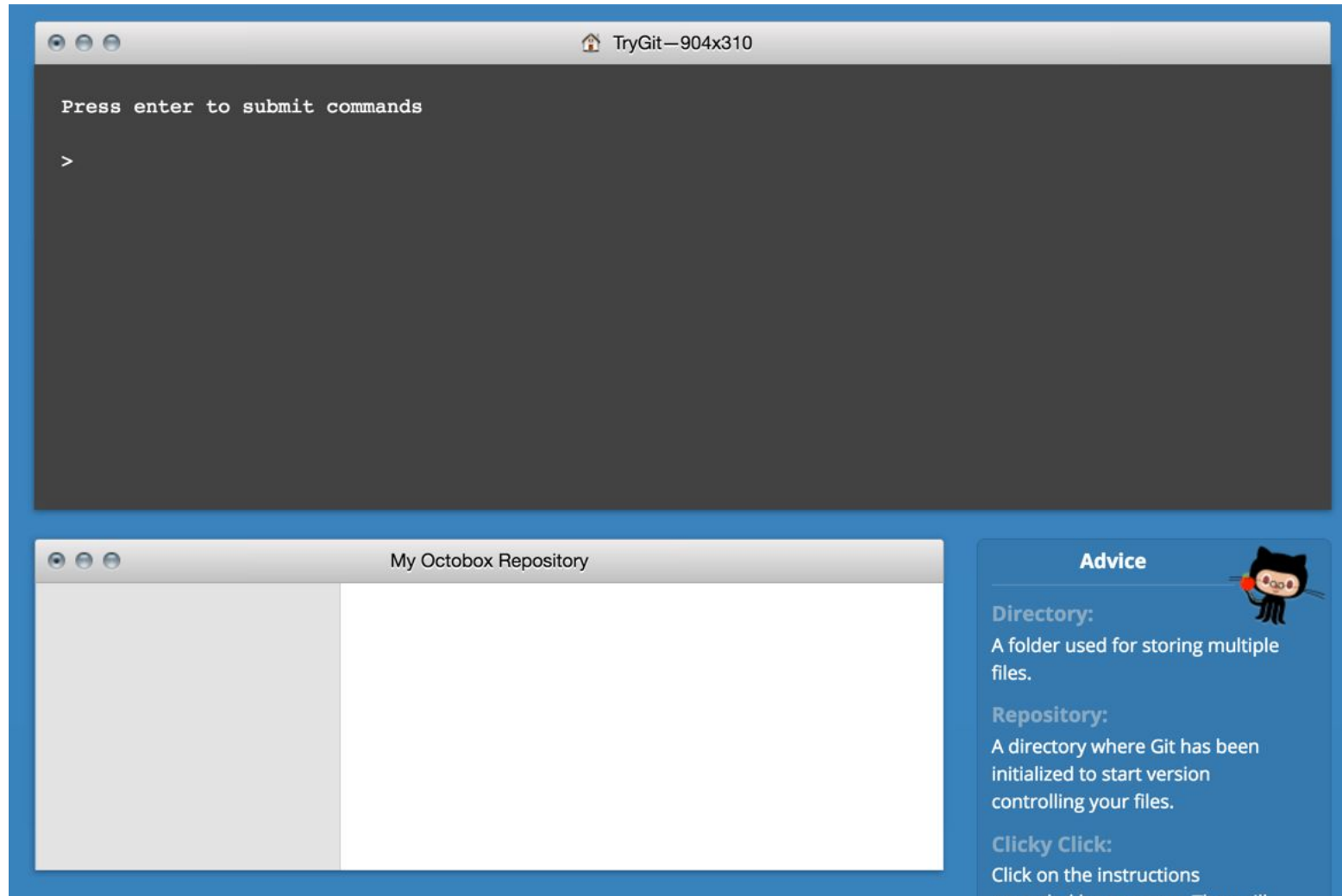
Version Control System



- Open Source Software (OSS)
- Immutable
- Fast
- Efficient
- Distributed
- Backbone of developer world....
 - Almost every OSS project on the world is on github (or gitlab)
 - Any job involving code will require you to know git

Git

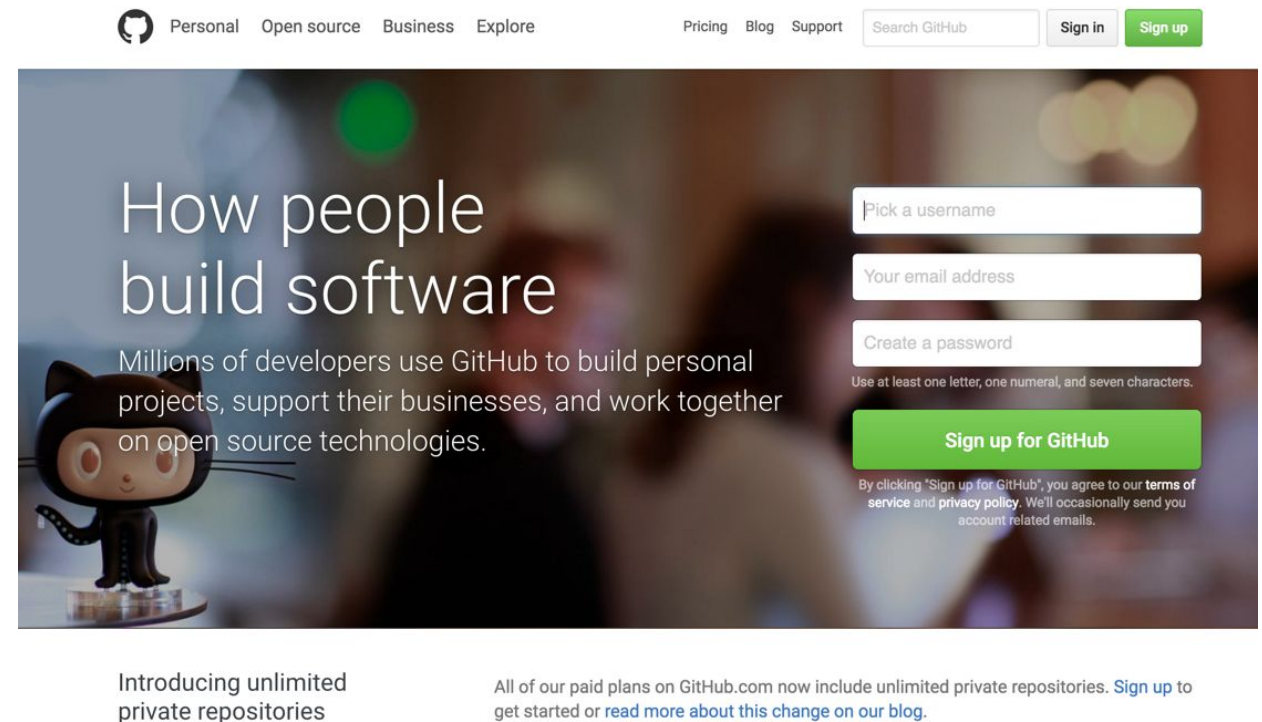
<https://try.github.io/>



Git

GitHub

- Create an account
- Make a repo (call it “python_class”)
- Clone it to c9
- Copy `test.py` to the repo
- Add, commit, push



Issue Tracking and Pull Requests (PRs)

- Make an issue
- Make a branch
- Fix the issue
- Pull the branch
- Make a PR
- Merge the PR

Tools

Jupyter

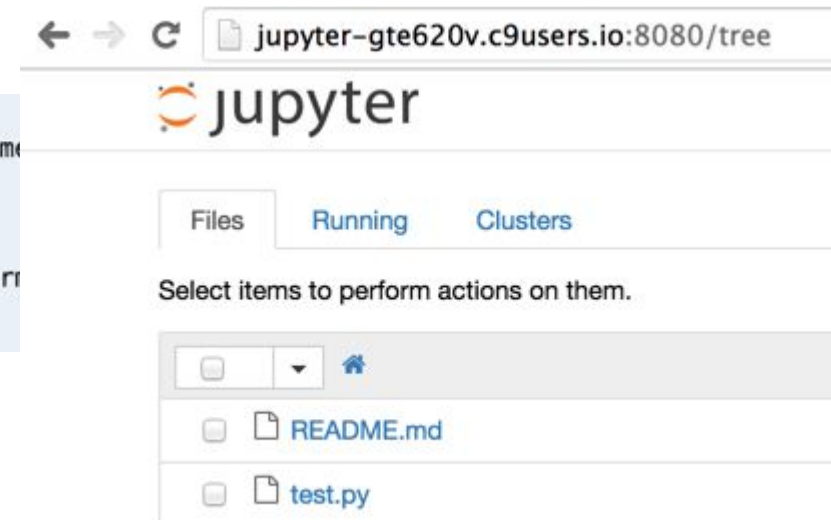
Jupyter

Start Jupyter

- run `jupyter-notebook --ip=0.0.0.0 --port=8080 --no-browser`
- Open new tab
 - Click on the link in the terminal

`http://0.0.0.0:8080/?token=2b6727cfe1f08710181d5d9235cdd86cbd679d1cb6f14edf`

```
gte620v:~/workspace $ jupyter notebook --ip=0.0.0.0 --port=8080 --no-browser
[I 19:59:11.754 NotebookApp] Writing notebook server cookie secret to /home/ubuntu/.local/share/jupyter/runtime
[I 19:59:11.920 NotebookApp] Serving notebooks from local directory: /home/ubuntu/workspace
[I 19:59:11.920 NotebookApp] 0 active kernels
[I 19:59:11.920 NotebookApp] The Jupyter Notebook is running at: http://0.0.0.0:8080/
[I 19:59:11.920 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirm
```



Jupyter?

Why Python Files?

- Completely portable
- Easy to organize, lint and test
- Is required for production

Why Jupyter?

- For Data Science
- Easy to connect code to output plots and descriptive text
- Reproducible science
- [Great learning tool](#) ([books](#))

“Many scientists in computation-intensive fields write code in an iterative and piecemeal fashion as each analysis reveals new insight and spins off multiple lines of inquiry. Keeping track of the different versions of code that produce various figures, and linking those files with explanatory notes, is a headache.”



Jupyter

Notebook Repo

- Clone course repo: `git clone git@github.com:gte620v/PythonTutorialWithJupyter.git`
- Start with “A quick tour of Jupyter Notebooks.ipynb”
- Ipython under the hood
 - Tab completion
 - Docstring preview
 - Running bash and other languages
 - Special functions
 - Plotting to the notebook

Python

Coding

Python

Coding Basics

- Lists
- Dictionaries
- Loops
- Conditions
- Functions
- Modules

Python

Pandas

Pandas

Working with Data

- Entity Resolution ([blog post](#))
 - Requires a package I wrote at <https://github.com/gte620v/pyd3netviz>
 - We need to clone that repo and then pip install the package
 - `git clone git@github.com:gte620v/pyd3netviz.git`
 - `cd pyd3netviz`
 - `pip install .`
 - Now we can open the notebook in the examples directory