Computational Fluency Short Course

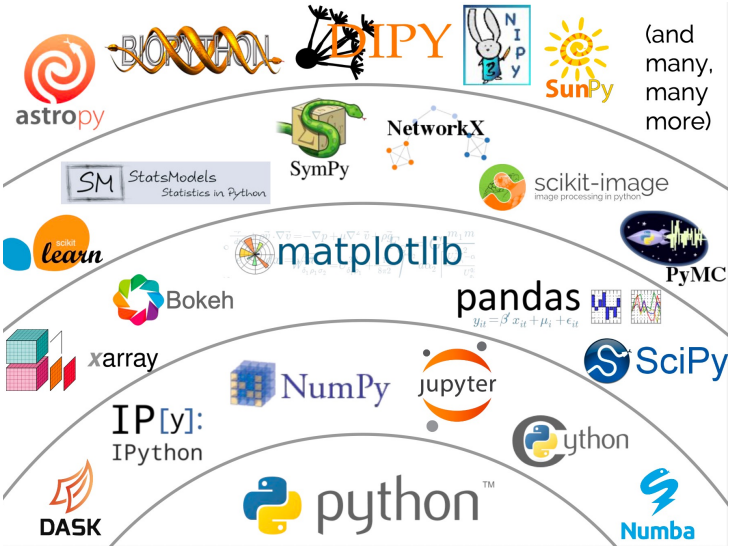
Virtual Environments and Package Managers

Jason Ritt
jason_ritt@brown.edu
Scientific Director of Quantitative Neuroscience

ROBERT J. & NANCY D. CARNEY
INSTITUTE FOR BRAIN SCIENCE
BROWN UNIVERSITY

https://github.com/brownritt/cfsc25

The boon of package management and virtual environments



We often expand the capability of our code by reusing existing code --- either our own or someone else's.

How do we keep track of all these code sources?

How do we make sure they play nice together?

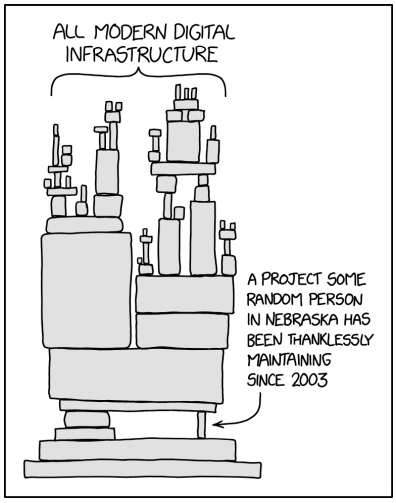
How can others reproduce our results using both our code and the exact packages that came from others?

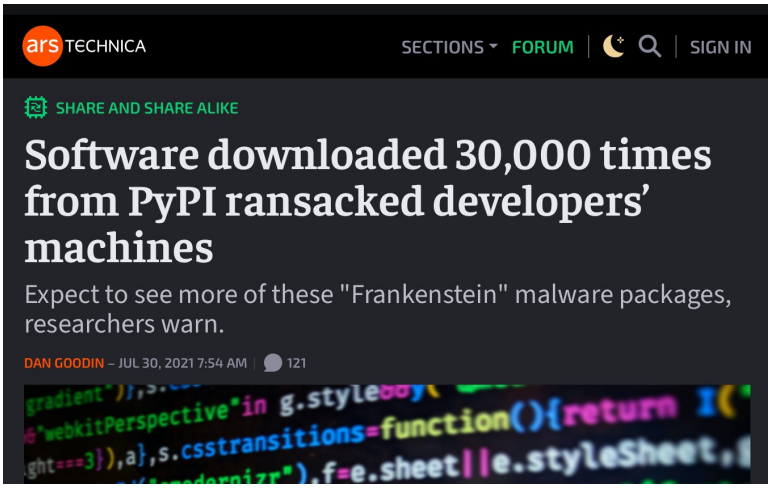
https://speakerdeck.com/jakevdp/the-unexpected-effectiveness-of-python-in-science

Package repositories: a cursed blessing

How do we know borrowed code works correctly?

How do we know it is safe?





https://xkcd.com/2347/
https://arstechnica.com/gadgets/2021/07/malicious-pypi-packages-caught-stealing-developer-data-and-injecting-code/

Name resolution

When code is coming from many places, it becomes quite likely that the same name is used many times for unrelated bits of code.

For example, how many packages have a `plot` function? How many functions use a variable named `x`?

There are two typical ways to use "name resolution" to handle this situation:

- ***Search paths***: the kernel searches an ordered list of places to look for a matching name; first found is first executed
- ***Namespace*** specification: the code says explicitly where to look, e.g. `df.plot` instead of `plt.plot`

Both are related to ***scope***, the idea that names have meaning only within some context specified by rules.