Programming Workshop

12 September 2016

Outline

- Why Linux?
- The shell:
  - Why use it?
  - What is it?
  - Example shell use
- Programming language overview -- Why Python?
- Linux setup discussion

Why Linux?

1. Enormous community
2. Easy to setup build environment
3. Free!
4. Universal
5. Lightweight without loss of features
6. Forces a more complete understanding of tools

Made by programmers for programming.

Programming in Shell vs GUI

GUI programming:

- Slow
- Resource-heavy
- Poorly supported
- Different between platforms
- Too helpful

Shell programming:

- Fast
- Lightweight
- Enormous community
- Largely universal
- Forces self-reliance

The Shell - What is it?

- Interface between user and hardware
- Direct access to part of OS called kernel
- Accessed via a terminal
- Most common terminal language is Bash

Example Bash Commands

ls : list everything in current directory

cd .. : go up one directory

cd /path/ : change directory to path

mkdir name : make directory called “name”

rm name : remove object called “name”

Programming Options Overview

- **GUI-Based Simulation**
  - SolidWorks, Autodesk, Ansys, Inventor, etc.
- **Math Languages**
  - Mathematica, Maple, MATLAB, Octave, ROOT, etc.
- **Programmatic Languages**
  - C, C++, Java, Python, FORTRAN, Assembly, HTML, etc.

GUI-Based Simulation

- Great for physical modeling
- Gracefully handle enormously complex, relational calculations
- VERY resource-heavy
- Platforms are generally similar but non-communicable
- Often costly
- Applications are limited based on software, hardware, physicality, etc.

Math Languages

- Great for quick symbolic solving
- Useful for quickly generating images – especially dynamic or interactive images
- Often very language-unreadable (complicated and involved syntax)
- Languages are often highly dissimilar
- Depending on platform, can be resource-heavy
- Learning curve is often steep
- Applications are limited by capabilities of platform

Programmatic Languages

Enormous variation.

- Some are language-readable, some aren't.
- Some run quickly, others don't.
- Some languages are easy to learn. Others aren't.
- Some languages make it easy to solve math-ey problems. Others make it easy to form GUIs. Some are better for recursive problems. Others work great in parallel.

And on and on and on…

Programming Languages

High-Level languages:

- C++, Java, HTML, Python, …
- Language-readable
- Approachable
- Versatile
- Quick to program
- Easy (ish) to debug

Functional languages:

- Haskall, C, Miranda, …
- Generally not readable
- Fast runtime, especially for math-ey problems
- Steep learning curve (directly logic-dependent)
- Often difficult to debug

So Why Python?

- Easy to learn
- Easy to write in (thus quick)
- Readable, force-formatted syntax
- Powerful, clever logic options (ex: list comprehension)
- Abundant package options:
  - GUI-formation
  - Database management
  - Image output (interactive and static)
  - And so, so much more!
- Moderately fast to run
- Great for math (both symbolic and discrete)

So Why Python? (cont’d)

- Moderately fast to run
- Great for math (both symbolic and discrete)
- High-level enough to be approachable, low-level enough to still give you adequate control
- Very commonly used in physics, software development, and engineering (resume fodder)
- **Enormous** community support

Questions?

Testing Linux Setup

“Homework”

(try not to look too excited)