Software Management

easy, apt peasy!

Running Software in Linux

- In Linux/Unix there is no “Windows Registry” or “add-remove programs” to keep track of where the programs are located, where to put shortcuts, or what files, libraries, or other decencies are required.
- To run any software:
  - Software source code can be obtained and compiled on your system
  - You can manually check to make sure all dependencies for the program are also available on your system and install separately as needed
  - Binary files can run directly on your system from anywhere given that the user that is running the executable has +x (execute) permission on that file

Software Repositories (repo)

- A location where software packages are stored/distributed
  - Can be local, on removable media, or on internet/network location
- Can be for individual software programs or entire operating systems
- Uses various forms of table of content or meta-data to provide a list of available packages and versions
- Operators/creators of such repositories typically provide a package management system
- Example of additional Ubuntu repos:
  - Universe (community driven open-source packages)
  - Multiverse (community driven, but with licensing issues that prevent them from being distributed with a free operating system)
  - 3rd party repos, such as Brave-browser: see here https://brave-browser.readthedocs.io/en/latest/installing-brave.html

Package managers

- Using package managers make installing, managing, updating, and removing programs easy
- Have a specific format and can retrieve software packages from repositories (example .deb, or .yum, or .rpm)
- Can keep track of all files and dependencies belonging to a software package in their own database
- Contain scripts (automation) to install, remove, and update software packages and their files
- Similar to an “App Store” on a modern mobile OS, except it has been around for much longer

Package Management in Ubuntu

- See documentation: https://ubuntu.com/server/docs/package-management
- dpkg (Debian Package manager) ← Backend (useful for scripting and performing tasks manually)
- APT (Advanced Packaging Tool) ← interactive, easy to use CLI front-end, not intended for scripting

APT

- List of repositories are in this file: `/etc/apt/sources.list`
  - You can view, modify, and remove repositories by editing that file
- You can review the log of activity: `/var/log/dpkg.log`
- Typical tasks:
  - `sudo apt update`
    - Updates local package index by pulling new index from package repositories
  - `sudo apt install <package name>`
  - `sudo apt remove <package name>`
  - `sudo apt upgrade`
    - Upgrade the local packages from the repository

Debian Package Management (dpkg)

- Typical usage:
  - `#: dpkg -l`
    - Lists all the packages in the package db (installed or not)
    - Example of better usage scenario: `dpkg -l | grep <package name>`
  - `#: dpkg -L <package name>`
    - Lists all the files a package has installed on the system
  - `#: dpkg –S <file name>`
    - Tells you which package may have put this file here!
  - `#: dpkg -i <.deb file name>.deb`
    - This installs a package file you have downloaded yourself
  - `#: dpkg --reconfigure <package name>`
    - This is used to run the installation wizard or the specific package setup program again
      - Example: `sudo dpkg --reconfigure mysql      # ← this will run the configuration wizard for myql in case you missed it the first time during the initial installation process`