Developing *art* Experiment Code with Docker, Xcode, and Psychic Easter Bunny Dolphins

Adam Lyon/Fermilab SCD & g-2

***art* Workshop**

17 June 2016

My background

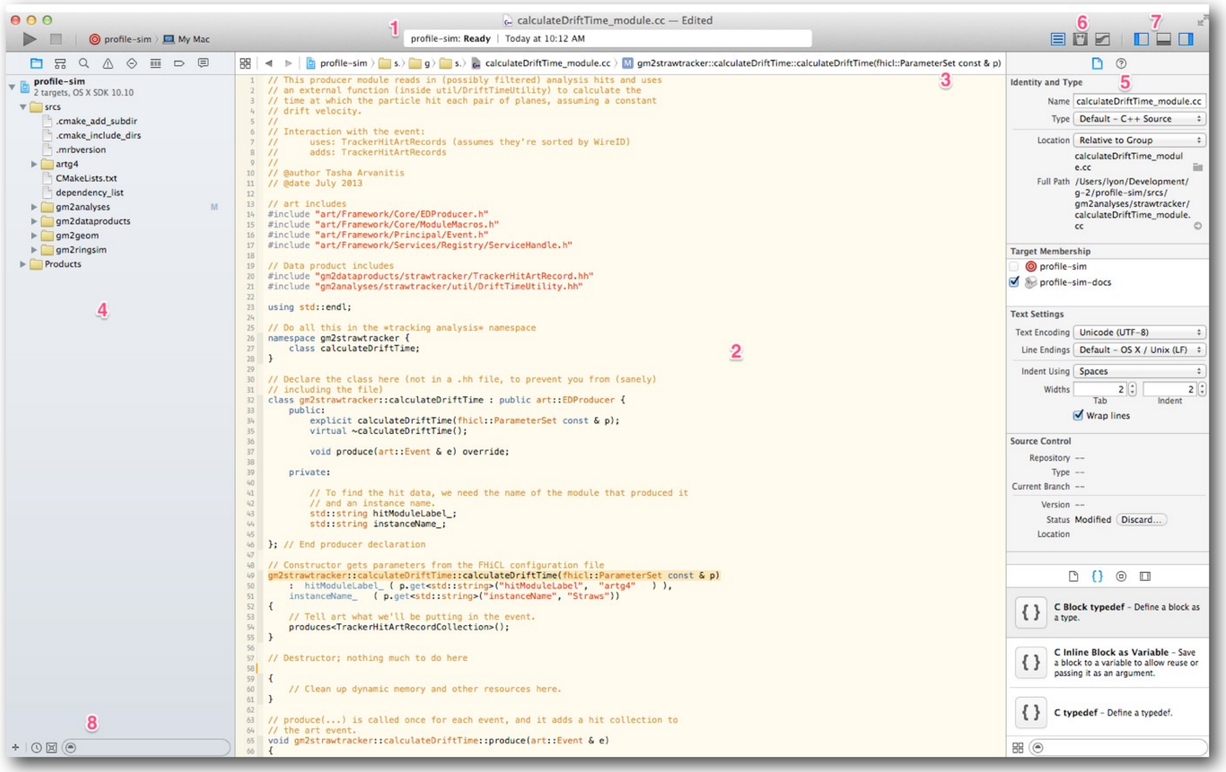
- Scientist on Muon g-2 and SCD Quadrant Head
- When I develop code, I use my Mac
  - It's powerful (8 cores lookin' for stuff to do)
- It's fast (SSD: less spin == less wait)
- It's nearby (ssh dont.need.no.ssh)
- It's got Xcode (avoid emacs finger injuries)
- Stuff works!

Xcode

See E989 Note 51: Developing g-2 Code on your Mac

3.1 QUICK TOUR

The picture at the right is a general display from *Xcode* editing source. Refer to the numbers for explanations. Hovering the mouse cursor over something will often give you help on that item.



1) **Activity bar.** This bar shows any activity from *Xcode* (building, indexing, debugging)

2) **Standard Editor.** This is the main editor window for source code.

3) **Jump bar.** Clicking parts of the jump bar allows you to quickly navigate to different parts of the code or other files. You can filter by clicking and then typing. Clicking The left and right triangles allow you to cycle backwards and forwards through viewed code. The four square thing to the left of the triangles is *very* useful. It is a context sensitive list of associated files (e.g. the counterpart file, which is the .h or .cc).

4) **Navigator bar.** Currently showing the *project navigator*. Note the "M" indicates that a file in that directory was modified and has not been committed to git.

5) **Utility bar.** Currently showing the file inspector. I find this bar less useful.

6) **Editor chooser.** Brings up different editors. The version editor (right most) is very useful.

7) **Bar chooser.** Can make the navigator, debug, and utility bars disappear and re-appear.

8) **View restrictor.** Restricts what you see in the navigator view.

Fermilab

My problem (well, one of them)

- Things were great until

- Introduced System Integrity Protection (SIP)

```
mac-124553:~ lyon$ export A_VARIABLE="pizza" ; echo $A_VARIABLE
pizza
mac-124553:~ lyon$ export DYLD_LIBRARY_PATH=Something ; echo $DYLD_LIBRARY_PATH
Something
mac-124553:~ lyon$ bash # Start subshell
mac-124553:~ lyon$ echo $A_VARIABLE # This works
pizza
mac-124553:~ lyon$ echo $DYLD_LIBRARY_PATH # SIP in action

mac-124553:~ lyon$
```

SIP blocks propagation of DYLD_LIBRARY_PATH to subshells

UPS relies on DYLD_LIBRARY_PATH for relocatability

If DYLD_LIBRARY_PATH doesn't work…
Builds don't work
art doesn't work
… without very nasty hacks

Fermilab

What to do?

- Turn off SIP? – Can do that, but I think that's a crutch.
  Future version of MacOS may not allow
  (rumor says can't turn off SIP in Sierra)
- Downgrade to Yosemite – Yuck
- Change UPS or use something else – See Spack talk for a direction along these lines
- Use Linux – If you can't beat 'em, join 'em
  - Throw away the Mac – noooooooo
  - Use a Scientific Linux Virtual Machine – Virtual box/Vagrant
  - Use Docker

What is ~~Docker~~ a Virtual Machine?

- Your host machine - (e.g. Mac Laptop) emulates a computer with a guest OS
- The guest OS may be anything - Windows, a different MacOS, Linux
- VirtualBox is an Open Source application that provides such Virtual Machines
- Vagrant is a nice application for configuring and provisioning VMs

- You typically end up with an "empty" machine that you must configure and provision (populate with system libraries, executables, etc - yum install)

- Most VM systems allow for network isolation with specific port forwarding, isolated user-space, sharing certain directories with the host

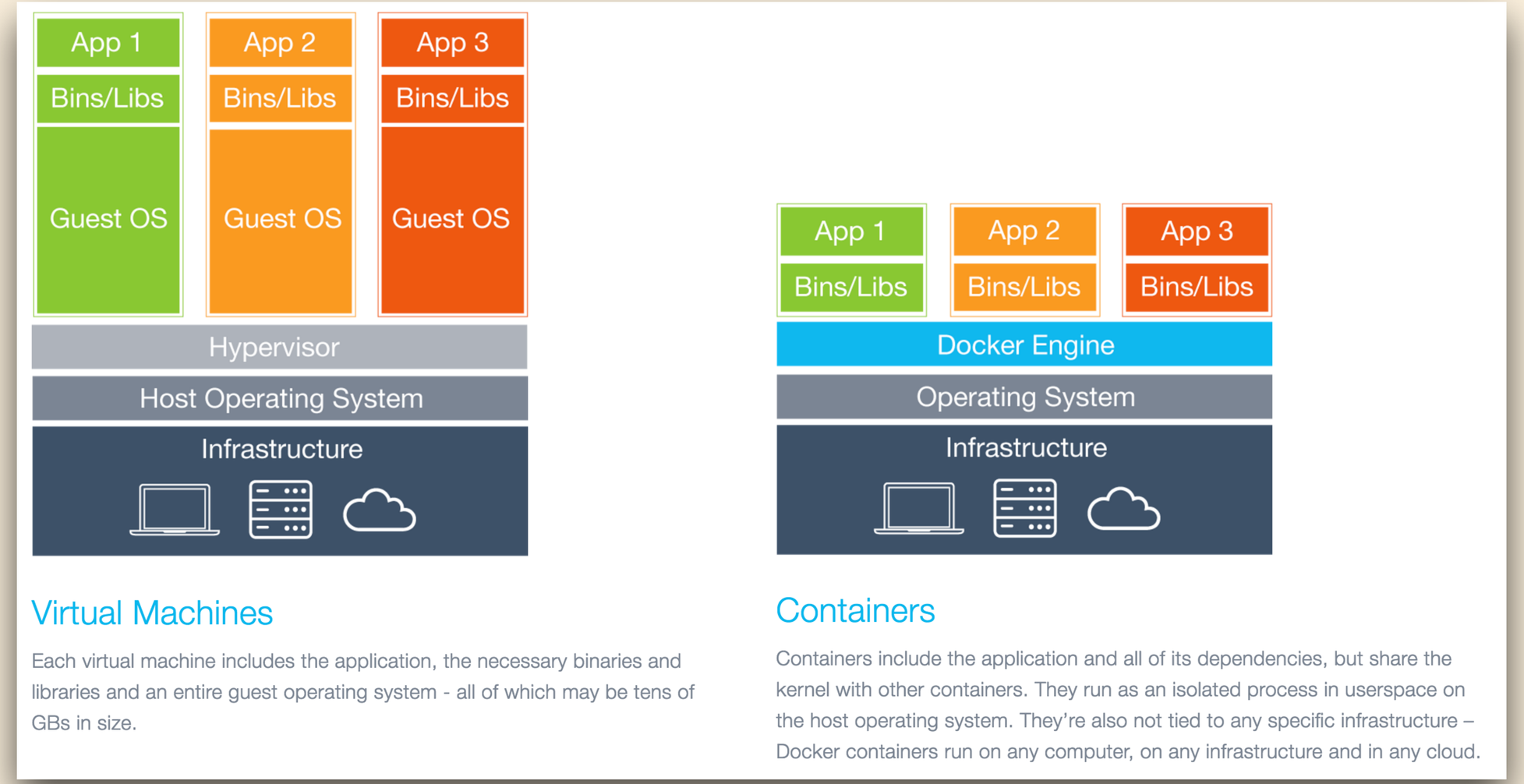
Another problem – clutter

- Trying out applications/libraries often requires installing lots of things in your system
- May take many tries to get things right
- You end up with a big mess

- Good example is trying to install jupyter notebooks with an *art* compatible Python
- What if you want to do more than one application?

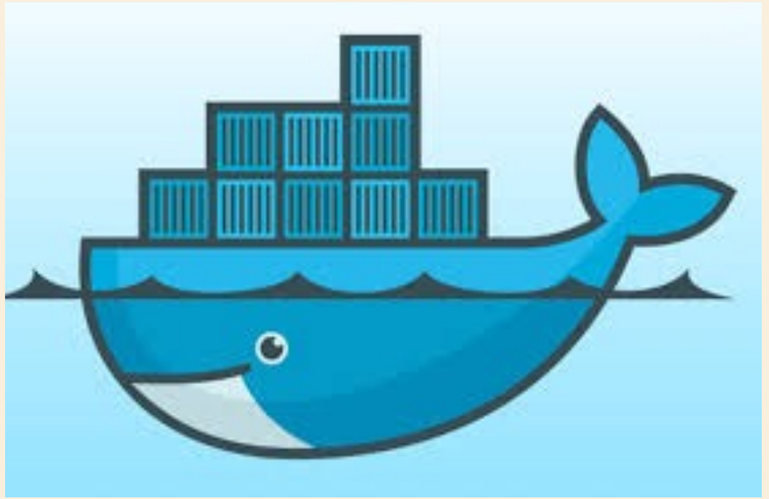
What is Docker (aside from more awesomeness)?

- From https://www.docker.com/what-docker … this will just scratch the surface



Fermilab

How does docker really work



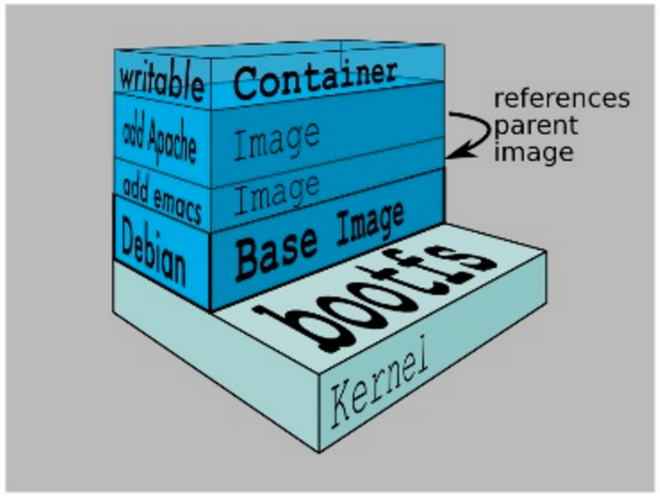
- With some help, a modern Linux kernel can run any flavor of Linux

- Docker allows for simultaneous running of many Linux instances on the same host sharing the host's kernel (which you don't care about, because it just works)

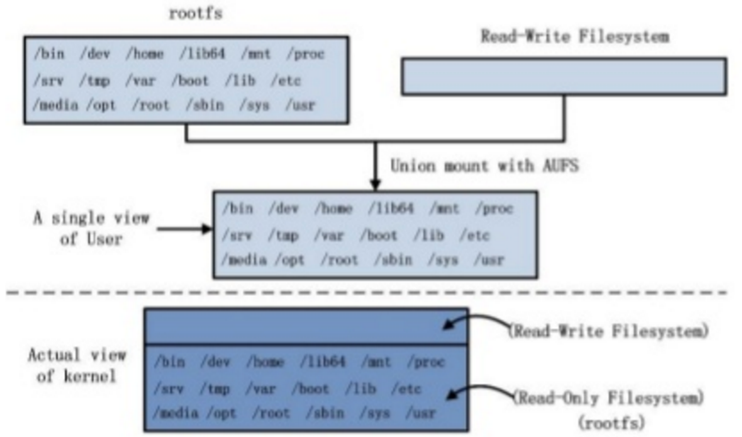
- Each instance or "container" has isolated user-space and system files with different networking options
  - o A container may live within the host's network space - generally not a good idea
  - o Containers may live on an internal network isolated from the host with port forwarding
  - o Multiple internal networks may be created for specific container groups
  - o Enormously flexible - and thus a bit confusing

- The "docker way" is to make a container per application

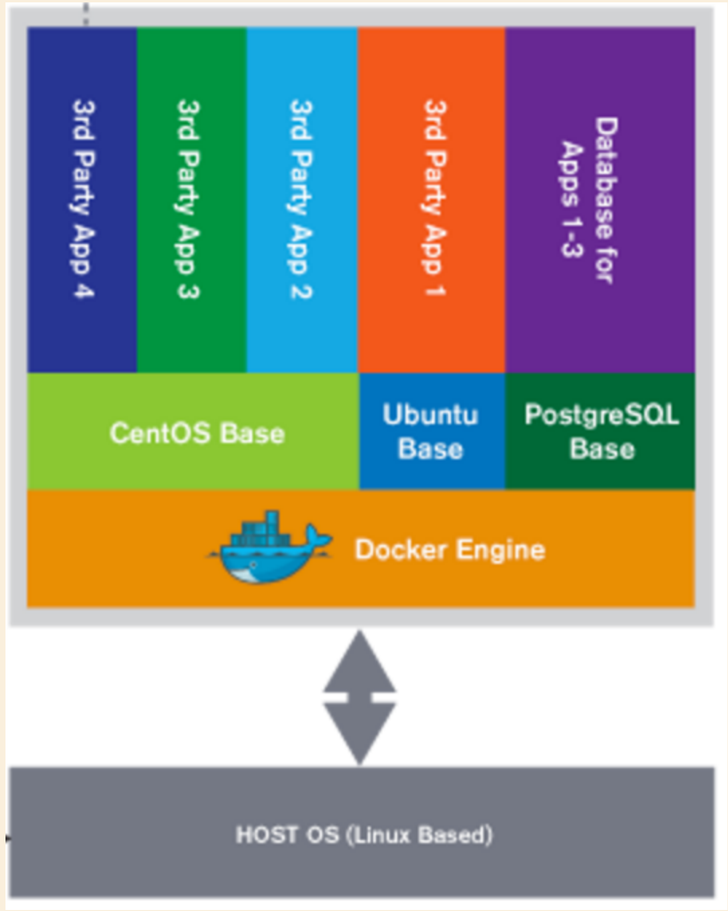
Docker containers and images



- Each layer of the FS is mounted on top of prior layers
- The first layer is the base image
- Current base images include debian, ubuntu, busybox, fedora, cent os, etc
- Each read-only layer is called an image (A layer is just a collection of files and folders!)
- The top layer is the only modifiable layer - it's termed the container



http://www.slideshare.net/Laynepeng/docker-introduction-48188539



http://www.asigra.com/solutions/docker-container-environments

Image/Container paradigm allows for sharing

Fermilab

Docker solves problems for “Development and Operations”

- Collaboration between developers and IT - a big deal in industry
- DevOps addresses the full environment for…

DevOps toolchain [edit]

Because DevOps is a cultural shift and collaboration between development, operations and testing, there is no single DevOps tool, rather a set or “DevOps toolchain” consisting of multiple tools.[14] Generally, DevOps tools fit into one or more of these categories, which is reflective of the software development and delivery process:[15][16]

- Code – Code Development and Review, continuous integration tools
- Build – Version control tools, code merging, Build status
- Test – Test and results determine performance
- Package – Artifact repository, Application pre-deployment staging
- Release – Change management, Release approvals, release automation
- Configure – Infrastructure configuration and management, Infrastructure as Code tools
- Monitor – Applications performance monitoring, End user experience

Though there are many tools available, certain categories of them are essential in the DevOps toolchain setup for use in an organisation. Some attempts to identify those basic tools can be found in the existing literature.[17]

Tools such as Docker (containerization), Jenkins (continuous Integration), Puppet (Infrastructure as Code) and Vagrant (virtualization platform) among many others are often used and frequently referenced in DevOps tooling discussions.[18]

https://en.wikipedia.org/wiki/DevOps

- Docker enables replicable, portable, and scalable environments

But what's that you say? The Mac isn't Linux and so can't play?

- Good point - The Mac doesn't have a Linux Kernel, nor does Windows

- Docker has a very thin virtual machine (boot2docker) to provide a kernel
  See Docker Toolbox (boot2docker in Virtualbox) or HomeBrew

- Docker for Mac and Windows Beta
  - Aims for a more seamless experience (like having a Linux host – you aren't meant to notice the VM)
  - Very thin Xhyve VM - no VirtualBox needed
  - BUT - it's beta (now on 15th iteration) - need an invitation token

- With Docker ANY modern machine can host your environment
- We've used Docker images to run MicroBoone code at NERSC on a Cray

DEMOS

- I know, you're tired of looking at slides — DEMO TIME!

- Trying things out… See https://github.com/lyon-fnal/docker-gm2
- brew install socat [Install homebrew]

DEMOS

- Show images and containers
- Look at a Dockerfile
- setup_docker
- Development container
  - kinit, kx509
  - C-p C-q; docker attach
  - docker exec
  - docker stop, kill, rm
  - -v, —name=, —volumes-from=

- Xcode build

Demos

- Run RStudio
- Run Catalyst

Summary

- Docker allows running Linux flavors on your Mac
- Docker enables organization
- Docker gets you past SIP

- Xcode can deal with Docker with some hacks

- Docker is likely in our future - resistance is futile

- A nice way to work

#psychicEasterBunnyDolphins #awesome