Containers

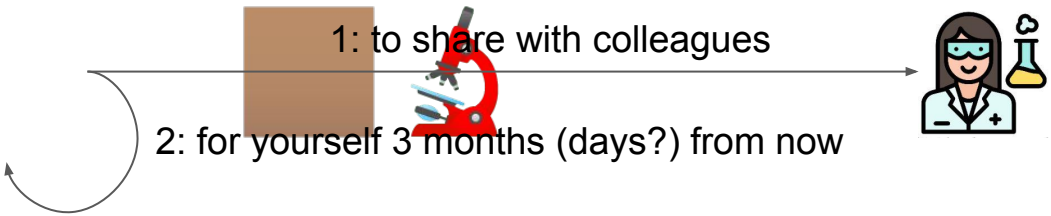
QLS 612 - July 2022

Sebastian Urchs - @s_urchs

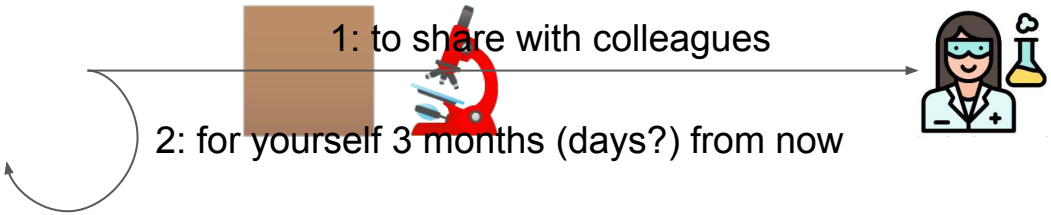
What we will talk about

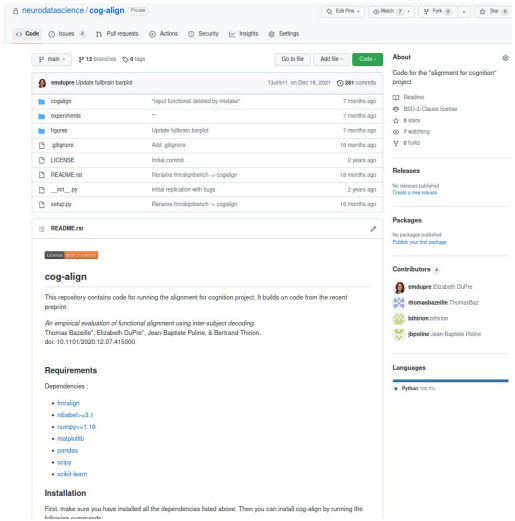
1. **Why** are containers useful for researchers
2. Python **virtual environments**
3. **Virtual Machines**
4. Containers with **Docker**
5. Using containers on supercomputers with **Singularity**

Document your software environment



Document your software environment





https://github.com/neurodatascience/cog-align

To install Python dependencies, we need:

1 - A Package manager to install things

pip

https://pip.pypa.io/en/stable/

To install Python dependencies, we need:

1 - A **package manager** to install things

pip

https://pip.pypa.io/en/stable/

2 - A list of dependencies to install

these are usually in one of:

- requirements.txt
- setup.cfg
- setup.py
- pyproject.toml

e.g. https://github.com/nilearn/nilearn/blob/main/setup.cfg

```
6  [options]
7  python_requires = >=3.6
8  install_requires =
9          joblib>=0.15
10         lxml
11         nibabel>=3.0.0
12         numpy>=1.18
13         pandas>=1.0
14         requests>=2
15         scikit-learn>=0.22
16         scipy>=1.5
17
```

To install Python dependencies, we need:

1 - A **package manager** to install things

pip

https://pip.pypa.io/en/stable/

2 - A list of dependencies to install

these are usually in one of:

- requirements.txt
- setup.cfg
- setup.py
- pyproject.toml

e.g. https://github.com/nilearn/nilearn/blob/main/setup.cfg

```
[options]
python_requires = >=3.6
install_requires =
        joblib>=0.15
        lxml
        nibabel>=3.0.0
        numpy>=1.18
        pandas>=1.0
        requests>=2
        scikit-learn>=0.22
        scipy>=1.5

```

3 - A **repository** of packages to install from

https://pypi.org/

To install Python dependencies, we need:

1 - A **package manager** to install things

pip

https://pip.pypa.io/en/stable/

2 - A list of dependencies to install

these are usually in one of:

- requirements.txt
- setup.cfg
- setup.py
- pyproject.toml

e.g. https://github.com/nilearn/nilearn/blob/main/setup.cfg

```
[options]
python_requires = >=3.6
install_requires =
        joblib>=0.15
        lxml
        nibabel>=3.0.0
        numpy>=1.18
        pandas>=1.0
        requests>=2
        scikit-learn>=0.22
        scipy>=1.5

```

3 - A **repository** of packages to install from

https://pypi.org/

Install

First make sure you have installed all th
can install nilearn by running the follow

```
pip install -U --user nilearn
```

To install Python dependencies, we need:

1 - A **package manager** to install things

pip

https://pip.pypa.io/en/stable/

2 - A list of dependencies to install

these are usually in one of:

- requirements.txt
- setup.cfg
- setup.py
- pyproject.toml

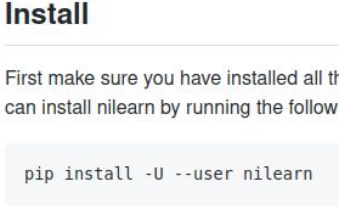
e.g. https://github.com/nilearn/nilearn/blob/main/setup.cfg

```
6  [options]
7  python_requires = >=3.6
8  install_requires =
9          joblib>=0.15
10         lxml
11         nibabel>=3.0.0
12         numpy>=1.18
13         pandas>=1.0
14         requests>=2
15         scikit-learn>=0.22
16         scipy>=1.5
17
```

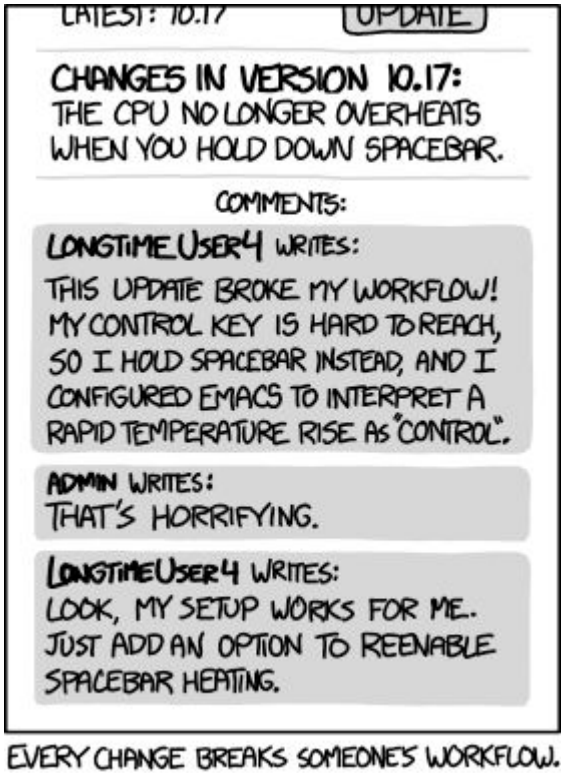
dependencies may have specific version requirements

3 - A **repository** of packages to install from

https://pypi.org/



Don’t use the same environment for all projects



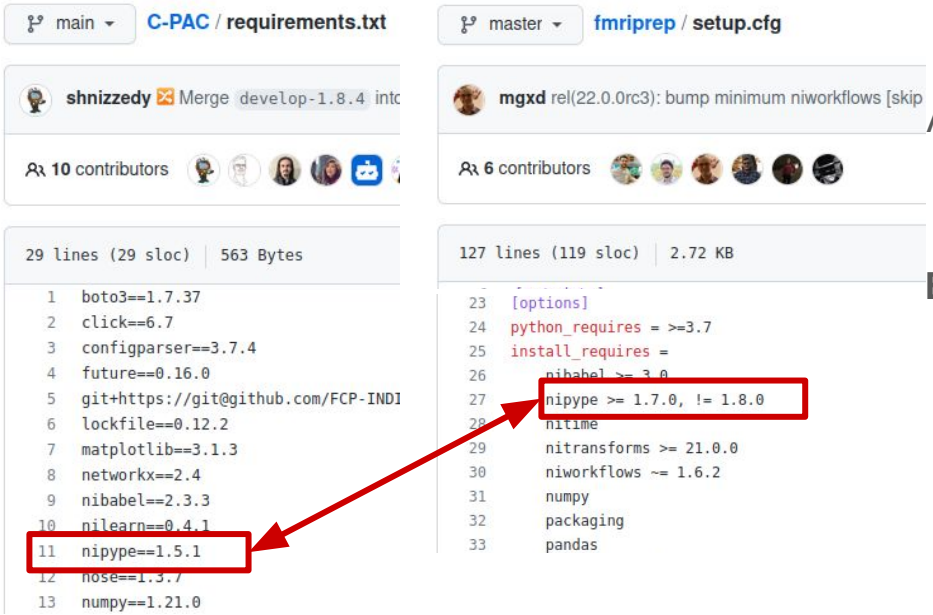
https://xkcd.com/1172/

Changing dependencies may do unexpected things

A. Your updated the dependencies of an existing project

Don’t use the same environment for all projects

Changing dependencies may do unexpected things



A. Your updated the dependencies of an existing project

B. **Two projects use the same environment but need different versions of some dependencies**

Do not install things into your system Python!

Common installation issues

Installing into the system Python on Linux

On Linux systems, a Python installation will typically be included as part of the distribution. Installing into this Python installation requires root access to the system, and may interfere with the operation of the system package manager and other components of the system if a component is unexpectedly upgraded using `pip`.

On such systems, it is often better to use a virtual environment or a per-user installation when installing packages with `pip`.

```
[surchs@marvin ~]$ which python
/usr/bin/python
[surchs@marvin ~]$ which pip
/usr/bin/pip
[surchs@marvin ~]$
```

https://docs.python.org/3/installing/index.html

Consider: a cake

Home · Cakes · Perfect Cream Cheese Pound Cake

Perfect Cream Cheese Pound Cake

Published by **Sally** on February 18, 2019 - **700 comments**

not sponsored, but I absolutely adore Nordic Ware Bundt pans.
10-12 cups of batter. **This one** is also gorgeous! 🙂

5 **Bake:** Bake the cream cheese pound cake at 325°F (163°C). Hal
the cake with aluminum foil to prevent over-browning.

6 **Cool, then invert:** Let the pound cool for about 2 hours in the
plate and cool completely before serving.

https://sallysbakingaddiction.com/cream-cheese-pound-cake/

Consider: a cake

Home · Cakes · Perfect Cream Cheese Pound Cake

Perfect Cream Cheese Pound Cake

Published by **Sally** on February 18, 2019 - **700 comments**

not sponsored, but I absolutely adore Nordic Ware Bundt pans.
10-12 cups of batter. **This one** is also gorgeous! 🙂

5 **Bake:** Bake the cream cheese pound cake at 325°F (163°C). Hal
the cake with aluminum foil to prevent over-browning.

6 **Cool, then invert:** Let the pound cool for about 2 hours in the
plate and cool completely before serving.

https://sallysbakingaddiction.com/cream-cheese-pound-cake/

Isolate environments to handle different requirements

How can I isolate Python environments?

venv — Creation of virtual environments ¶

New in version 3.3.

Source code: Lib/venv/

The venv module provides support for creating lightweight “virtual environments” with their own site directories, optionally isolated from system site directories. Each virtual environment has its own Python binary (which matches the version of the binary that was used to create this environment) and can have its own independent set of installed Python packages in its site directories.

See **PEP 405** for more information about Python virtual environments.

See also: Python Packaging User Guide: Creating and using virtual environments

https://docs.python.org/3/library/venv.html#module-venv

Python **V**irtual **Env**ironment: what is it, what does it do?

Creation of virtual environments is done by executing the command `venv`:

```
python3 -m venv /path/to/new/virtual/environment
```

1. makes a new **directory**
2. copies the system python interpreter there
3. creates a sub-directory to install dependencies

You can “activate” it by using the “activate” shell script

```
test-project/venv/              # Our environment's root directory
├── bin
│   ├── activate                          # Scripts to activate
│   ├── activate.csh                      # our project's
│   ├── activate.fish                     # virtual environment.
│   ├── easy_install
│   ├── easy_install-3.7
│   ├── pip
│   ├── pip3
│   ├── pip3.7
│   ├── python -> /usr/local/bin/python   # Symlinks to system-wide
│   └── python3 -> python3.7              # Python instances.
├── include
├── lib
│   └── python3.7
│       └── site-packages             # Stores local site packages
└── pyvenv.cfg
```

https://docs.python.org/3/library/venv.html

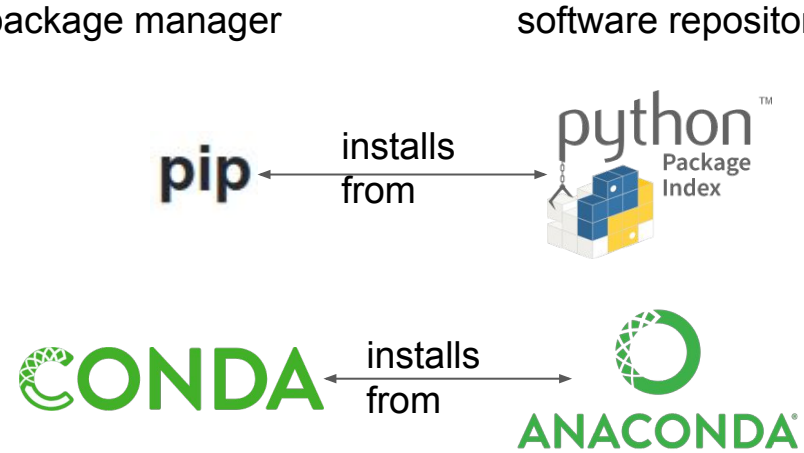
https://towardsdatascience.com/virtual-environments-104c62d48c54

VENV Demo

Python **VENV**: isolated python environments

- use **venv** to create python virtual environments from existing python
- a virtual environment is (almost) just a new **directory** to put dependencies in
- make a **new one** for each project
- **document** dependencies with requirements.txt or setup.cfg, …
- **install** your dependencies into the environment
- activate and **check** your environment before you use it

Conda: a convenient Python distribution



conda

- package manager for **Anaconda**
- **smarter than pip** for dependency resolution
- also creates virtual environments like **venv**
- **environment.yml** instead of requirements.txt

Don’t mix pip and conda in the same environment!

Anaconda

- curated distribution of Python and R (packages)
- data science focus
- includes **non-Python** binary dependencies
- **not as many** packages as PyPA / pip
- “channels” exist that are curated by others / OSS
- can have GUI, good to get Python on Windows

https://www.anaconda.com/products/distribution

but see https://docs.alliancecan.ca/wiki/Anaconda/en#Do_not_install_Anaconda_on_our_clusters

What can Python virtual environments not do?

Prerequisites

The HCP Pipelines have the following software requirements:

1. A 64-bit Linux Operating System
2. The FMRIB Software Library (a.k.a. FSL) version 6.0.2 or greater installed and configuration file properly sourced. FSL 6.0.4 is recommended.
3. FreeSurfer version 6.0 available at http://surfer.nmr.mgh.harvard.edu/fswiki/DownloadAndInstall/

   FreeSurfer 7.X is not currently supported due to poor quality surface reconstructions.

   NB: You *must* create and install a license file for FreeSurfer by visiting and submitting the FreeSurfer registration form.

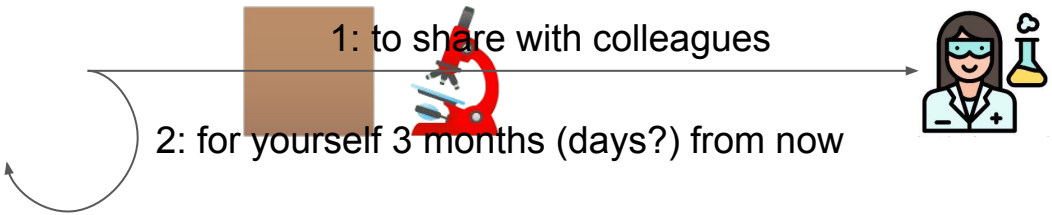
   NB: For now, FreeSurfer 5.3.0-HCP is still supported, but you must use the FreeSurferPipeline-v5.3.0-HCP.sh pipeline script instead.
4. Connectome Workbench version 1.4.2 or later

   The HCP Pipelines scripts use the `wb_command` which is part of the Connectome Workbench. They locate the `wb_command` using an environment variable. Instructions for setting this environment variable are provided below in the Running the HCP Pipelines on example data section.
5. The HCP version of gradunwarp version 1.1.0 (if gradient nonlinearity correction is to be done.)
6. MSM_HOCR v3.0/(github v1.0): https://github.com/ecr05/MSM_HOCR/releases This is the multi-modal surface matching algorithm used in MSMSulc and MSMAll.
7. FSL FIX v 1.0.6.14+: https://fsl.fmrib.ox.ac.uk/fsl/fslwiki/FIX sICA+FIX is used for cleaning spatially specific structured noise from fMRI data. Please see this page for supplemental instructions for sICA+FIX: https://github.com/Washington-University/HCPpipelines/blob/master/ICAFIX/README.md

https://github.com/Washington-University/HCPpipelines/wiki/Installation-and-Usage-Instructions#prerequisites

- Not all binary dependencies are on Anaconda
- Not everything is written in Python or R
- Your Operating System (**OS**) also has packages and a package manager
- The same version problems apply to these

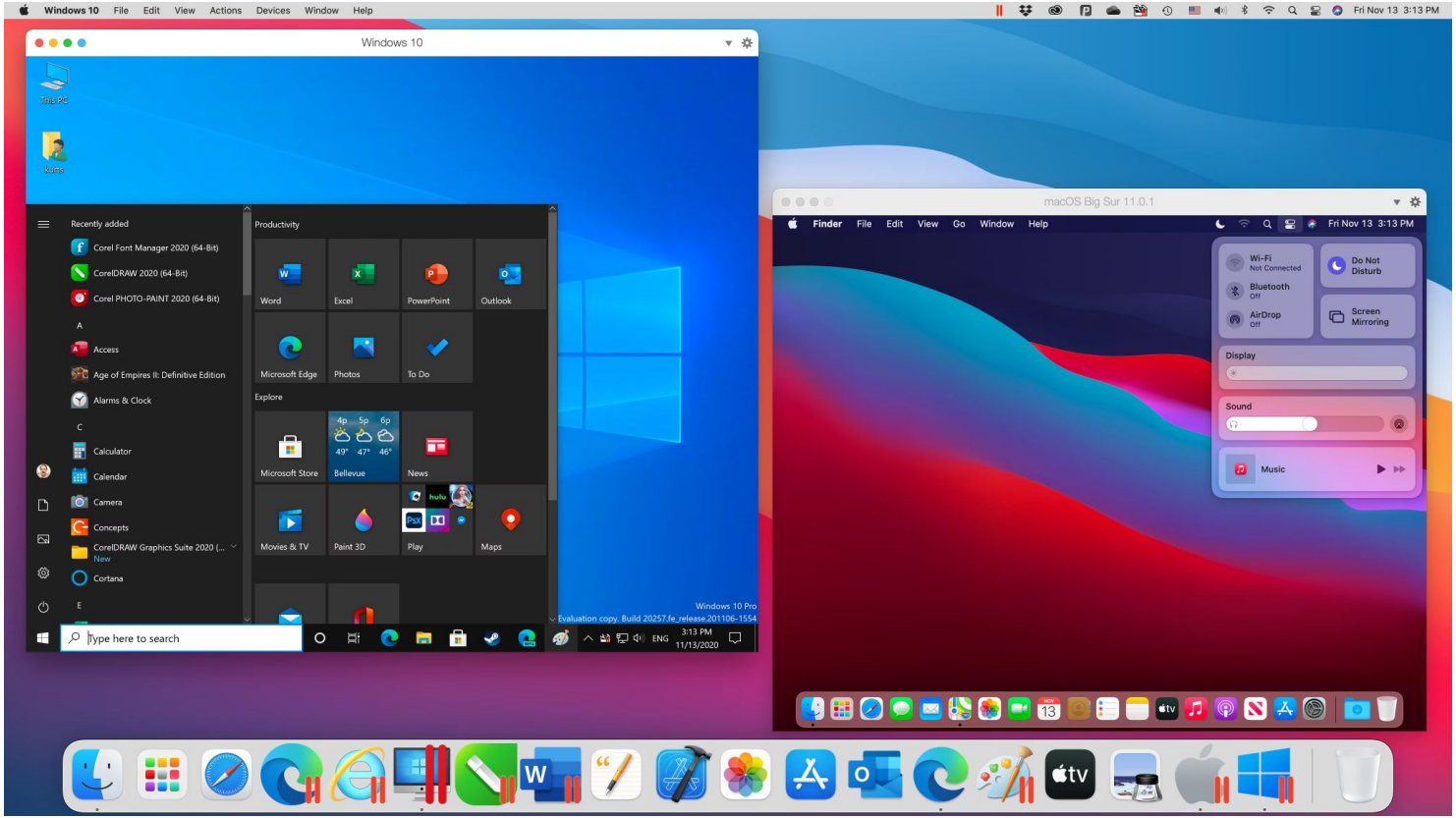
Need to capture the (entire) compute environment



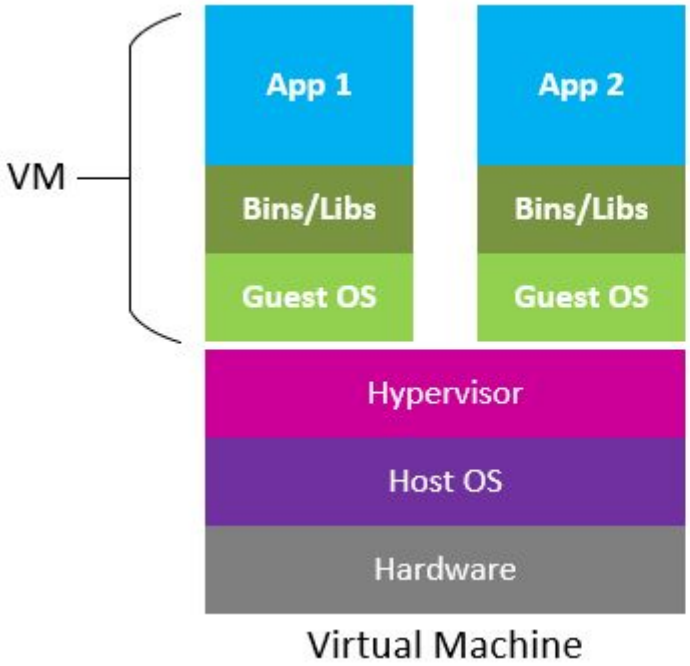
How can we have different OS environments?

Buy a lot of computers?

Virtual Machines: Let’s simulate the computers



Virtual Machine (VM)



What is it?

- entire computer hardware simulated (**hardware virtualization**)
- you can choose what hardware is exposed (CPU, RAM, hard-drive, …)
- the VM is fully isolated from the Host OS: you get to install any OS inside the VM

How do I make one?

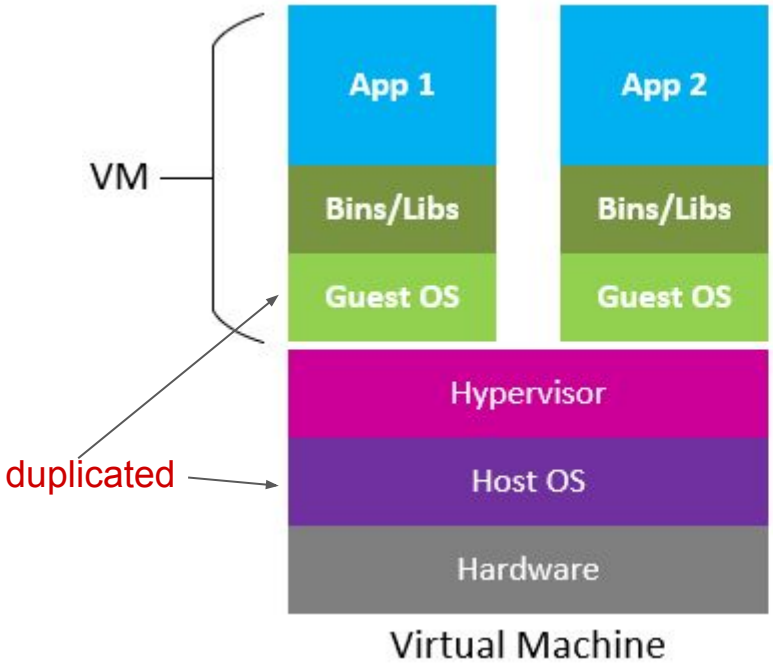
Get Hypervisor / Virtualization software

- VirtualBox (partly OSS)
- VMWare (commercial)
- Hyper-V (Windows)
- …

https://cloudblogs.microsoft.com/opensource/2019/07/15/how-to-get-started-containers-docker-kubernetes/

https://www.virtualbox.org/

Virtual Machine (VM)



Good:

- can run anything a computer runs (Windows, Linux, Hackintosh)
- can make a snapshot to share with others (Neurodebian used to have a VM)
- good way to test things across many systems
- full system isolation

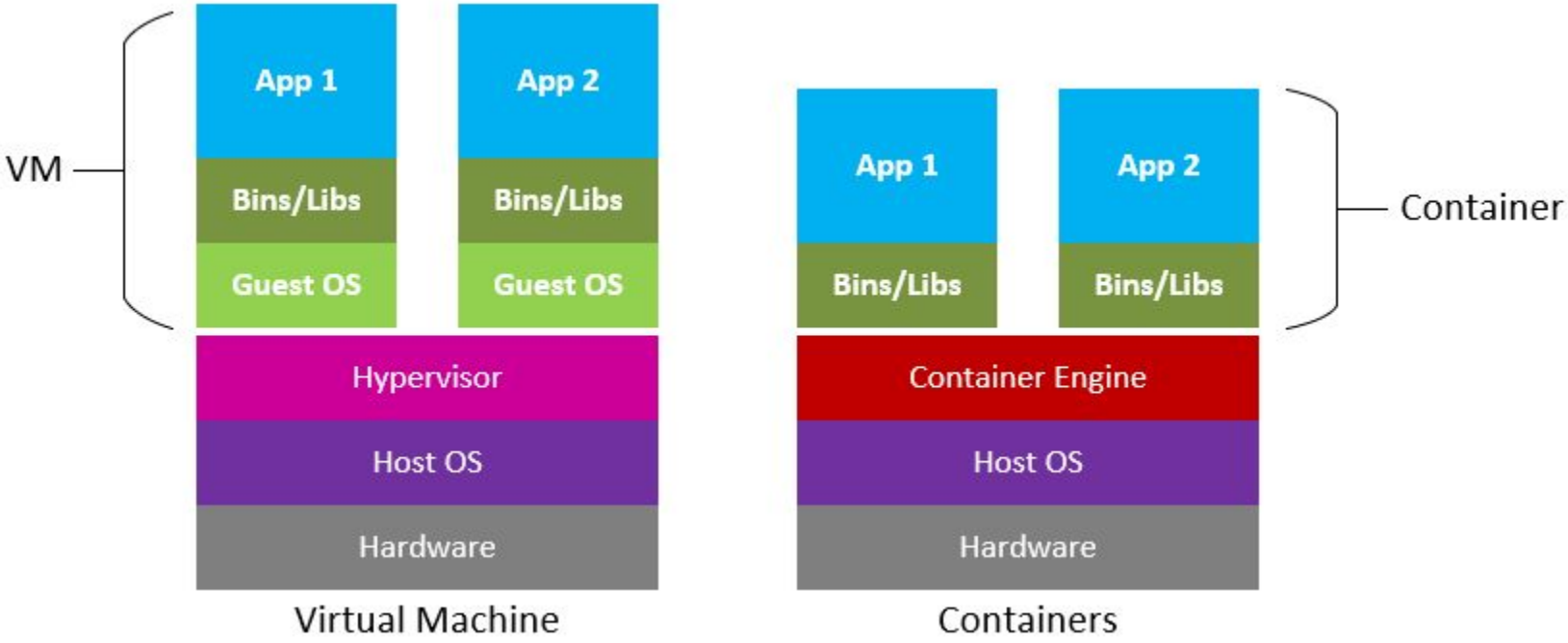
But:

- doesn't share resources with host -> BIG
- slow to start up, stop, resume
- cumbersome to configure for each project
- duplicates things (every VM needs OS / Kernel)
- no easy way to "get" and use VMs from other people
- you can't use it on a supercomputer

https://cloudblogs.microsoft.com/opensource/2019/07/15/how-to-get-started-containers-docker-kubernetes/

https://www.virtualbox.org/

Containers: let's all share the same kernel, but in boxes



https://cloudblogs.microsoft.com/opensource/2019/07/15/how-to-get-started-containers-docker-kubernetes/

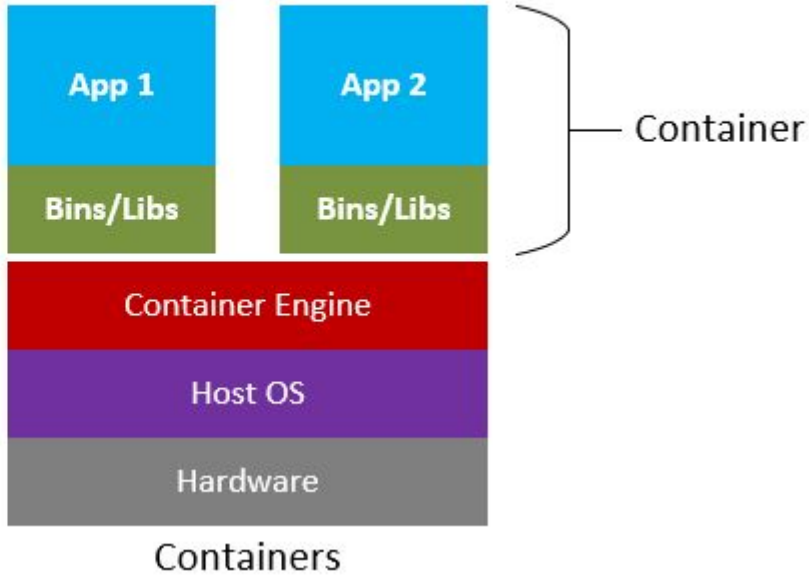
Containers

What are they

- isolated environments sharing the same kernel / OS -> (**OS virtualization**)
- from the inside, a container looks like a separate computer, can't see outside
- within each container, you can have your desired binary and library dependencies

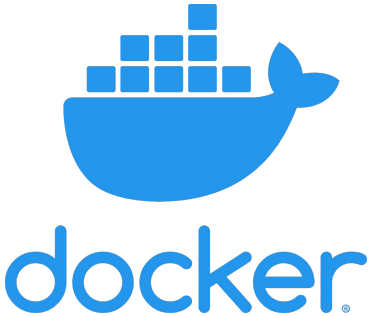
How do I make one

- use a container implementation
- **docker** is the most widely used
- Singularity is used on supercomputers



https://cloudblogs.microsoft.com/opensource/2019/07/15/how-to-get-started-containers-docker-kubernetes/

What is Docker



Docker Engine

- a command line program
- gets and builds Docker images and runs Docker containers

Docker Hub

- a website / web service
- a central repository to store and share Docker container images (commercial)
- other container image registries exist

https://docs.docker.com/engine/

https://hub.docker.com/

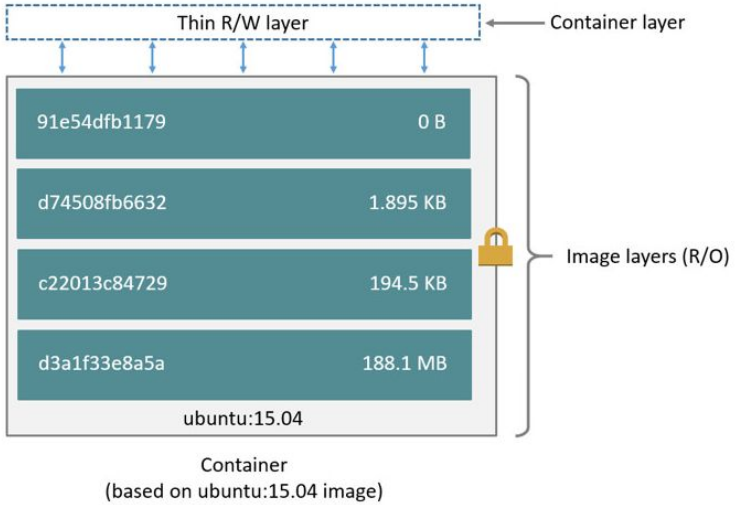
Docker **image** and **container**: what's the difference

Docker image

- a **read-only** snapshot of an environment
- organized in **layers**
- changing an image adds more layers
- can be stored on Dockerhub or as a file
- images can share identical layers
- can make your own with a **Dockerfile**

Docker container

- a **live instance** of a Docker image
- has a thin writable layer that dies with it
- **one image** can spawn **many containers**

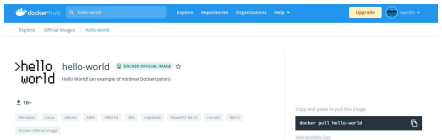


https://docs.docker.com/storage/storagedriver/

How can I get my own Docker container going

A: get an image **someone else** has shared

1. Find an image on Dockerhub



2. **docker pull** a local copy of the image

```
[surchs@marvin ~]$ docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:13e367d31ae85359f42d637adf6da428f76d75dc9afeb3c21faea0d976f5c651
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
```

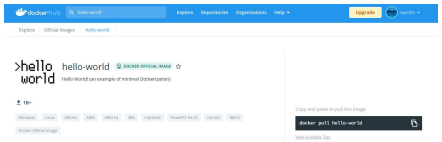
3. **docker run** a container of the image

```
[surchs@marvin ~]$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
```

How can I get my own Docker container going

A: get an image **someone else** has shared

1. Find an image on Dockerhub



2. **docker pull** a local copy of the image

```
[surchs@marvin ~]$ docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:13e367d31ae85359f42d637adf6da428f76d75dc9afeb3c21faea0d976f5c651
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
```

3. **docker run** a container of the image

```
[surchs@marvin ~]$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
```

B: **build your own** on top of an existing image

1. Find an image on Dockerhub to start with
2. Define your changes in a **Dockerfile**

```
# syntax=docker/dockerfile:1
FROM ubuntu:18.04
LABEL org.opencontainers.image.authors="org@example.com"
COPY . /app
RUN make /app
RUN rm -r $HOME/.cache
CMD python /app/app.py
```

3. **docker build** an image from the Dockerfile

```
[surchs@marvin docker_test]$ docker build -f Dockerfile -t mydocker_image .
Sending build context to Docker daemon  2.048kB
Step 1/6 : FROM ubuntu:18.04
18.04: Pulling from library/ubuntu
09db6f815738: Pull complete
Digest: sha256:478caf1bec1afd54a58435ec681c8755883b7eb843a8630091890130b15a79af
Status: Downloaded newer image for ubuntu:18.04
 ---> ad080923604a
Step 2/6 : LABEL org.opencontainers.image.authors="org@example.com"
 ---> Running in ca59864bac3a
Removing intermediate container ca59864bac3a
 ---> 785508425a2a
Step 3/6 : COPY . /app
```

4. use your local image

Let’s run a docker container

- Find an image we like: https://hub.docker.com/_/hello-world
- Take a look at it
- Pull the image
- Run the image:


```
[surchs@marvin ~]$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
```

Do something a little more useful

I don’t have conda installed on my system. But there is a Docker image. Let’s try!

- Find a Docker image: https://hub.docker.com/r/continuumio/miniconda3/
- Pull it
- Run it

Do something a little more useful

I don’t have conda installed on my system. But there is a Docker image. Let’s try!

- Find a Docker image: https://hub.docker.com/r/continuumio/miniconda3/
- Pull it
- Run it

Oh, nothing happened?

- Let’s run it interactively to take a look inside

```
[surchs@marvin ~]$ docker run --rm -ti continuumio/miniconda3:4.10.3-alpine
```

https://github.com/ContinuumIO/docker-images/blob/master/miniconda3/alpine/Dockerfile

Looking around inside a container

- no conda on my machine

```
[surchs@marvin ~]$
[surchs@marvin ~]$ hostname
marvin
[surchs@marvin ~]$ conda
bash: conda: command not found...
Install package 'conda' to provide command 'conda'? [N/y] n
```

- starting container changes the look of my terminal and the name of my computer

```
[surchs@marvin ~]$ docker run --rm -ti continuumio/miniconda3:4.10.3-alpine
(base) bash-5.1# hostname
820d61108a52
(base) bash-5.1#
```

- inside of the container I have access to conda

```
(base) bash-5.1# conda
usage: conda [-h] [-V] command ...

conda is a tool for managing and deploying applications, environments and pa
es.
```

Can I see files on my machine from inside the container?

```
[surchs@marvin docker_things]$ pwd
/home/surchs/Documents/docker_things
[surchs@marvin docker_things]$ ls
this_is_a_file_on_my_computer.txt
[surchs@marvin docker_things]$ docker run --rm -ti continuumio/miniconda3:4.10.3-alpine
(base) bash-5.1# pwd
/
(base) bash-5.1# ls
bin    etc    lib    media  opt    root   sbin   sys    usr
dev    home   lib64  mnt    proc   run    srv    tmp    var
(base) bash-5.1# cd /home/surchs/Documents/docker_things
bash: cd: /home/surchs/Documents/docker_things: No such file or directory
(base) bash-5.1#
```

outside (on host)

inside
(in container)

Can I write files inside my container

```
[surchs@marvin ~]$ docker run --rm -ti continuumio/miniconda3:4.10.3-alpine
(base) bash-5.1# ls
bin    etc    lib    media  opt    root   sbin   sys    usr
dev    home   lib64  mnt    proc   run    srv    tmp    var
(base) bash-5.1# cd home/
(base) bash-5.1# ls
(base) bash-5.1# touch this_is_a_file_in_my_container.txt
(base) bash-5.1# ls
this_is_a_file_in_my_container.txt
```

Do these files stick around?

```
[surchs@marvin ~]$ docker run --rm -ti continuumio/miniconda3:4.10.3-alpine
(base) bash-5.1# ls
bin     etc     lib     media   opt     root    sbin    sys     usr
dev     home    lib64   mnt     proc    run     srv     tmp     var
(base) bash-5.1# cd home/
(base) bash-5.1# ls
(base) bash-5.1# touch this_is_a_file_in_my_container.txt
(base) bash-5.1# ls
this_is_a_file_in_my_container.txt
(base) bash-5.1# exit
exit
```

first container instance

```
[surchs@marvin ~]$ docker run --rm -ti continuumio/miniconda3:4.10.3-alpine
(base) bash-5.1# ls
bin     etc     lib     media   opt     root    sbin    sys     usr
dev     home    lib64   mnt     proc    run     srv     tmp     var
(base) bash-5.1# cd home/
(base) bash-5.1# ls
```

second container instance

Could I please access my host files inside the container?

This is possible using and binding volumes:

```
[surchs@marvin docker_things]$ docker run -v /path/on/host:/path/inside/container my_image
```

- **-v**: the flag to bind a volume
- /path/on/host: the directory on my machine to expose to the container
- **:** divides host and container path
- /path/inside/container: where to expose the host directory

https://docs.docker.com/storage/volumes/

Could I please access my host files inside the container?

```
[surchs@marvin docker_things]$ pwd
/home/surchs/Documents/docker_things
[surchs@marvin docker_things]$ ls
this_is_a_file_on_my_computer.txt
[surchs@marvin docker_things]$ docker run -ti -v /home/surchs/Documents/docker_things:/inside_place continuumio/miniconda3:4.10.3-alpine
(base) bash-5.1# ls
bin           etc           inside_place  lib64         mnt           proc          run           srv           tmp           var
dev           home          lib           media         opt           root          sbin          sys           usr
(base) bash-5.1# cd inside_place/
(base) bash-5.1# ls
this_is_a_file_on_my_computer.txt
```

path on host

path in container

file on host

https://docs.docker.com/storage/volumes/

Could I please access my host files inside the container?

```
[surchs@marvin docker_things]$ pwd
/home/surchs/Documents/docker_things
[surchs@marvin docker_things]$ ls
this_is_a_file_on_my_computer.txt
[surchs@marvin docker_things]$ docker run -ti -v /home/surchs/Documents/docker_things:/inside_place continuumio/miniconda3:4.10.3-alpine
(base) bash-5.1# ls
bin           etc           inside_place  lib64         mnt           proc          run           srv           tmp           var
dev           home          lib           media         opt           root          sbin          sys           usr
(base) bash-5.1# cd inside_place/
(base) bash-5.1# ls
this_is_a_file_on_my_computer.txt
(base) bash-5.1#
(base) bash-5.1# echo "Hello from the other side" >> a_file_I_made_inside_a_container.txt
(base) bash-5.1# ls
a_file_I_made_inside_a_container.txt  this_is_a_file_on_my_computer.txt
(base) bash-5.1# cat a_file_I_made_inside_a_container.txt
Hello from the other side
(base) bash-5.1# exit
exit
[surchs@marvin docker_things]$ ls
a_file_I_made_inside_a_container.txt  this_is_a_file_on_my_computer.txt
```

path on host

path in container

file on host

file made in container

https://docs.docker.com/storage/volumes/

Could I please access my host files inside the container?

```
[surchs@marvin docker_things]$ pwd
/home/surchs/Documents/docker_things
[surchs@marvin docker_things]$ ls
this_is_a_file_on_my_computer.txt
[surchs@marvin docker_things]$ docker run -ti -v /home/surchs/Documents/docker_things:/inside_place continuumio/miniconda3:4.10.3-alpine
(base) bash-5.1# ls
bin           etc           inside_place  lib64         mnt           proc          run           srv           tmp           var
dev           home          lib           media         opt           root          sbin          sys           usr
(base) bash-5.1# cd inside_place/
(base) bash-5.1# ls
this_is_a_file_on_my_computer.txt
(base) bash-5.1#
(base) bash-5.1# echo "Hello from the other side" >> a_file_I_made_inside_a_container.txt
(base) bash-5.1# ls
a_file_I_made_inside_a_container.txt  this_is_a_file_on_my_computer.txt
(base) bash-5.1# cat a_file_I_made_inside_a_container.txt
Hello from the other side
(base) bash-5.1# exit
exit
[surchs@marvin docker_things]$ ls
a_file_I_made_inside_a_container.txt  this_is_a_file_on_my_computer.txt
[surchs@marvin docker_things]$ ls -l
total 8
-rw-r--r-- 1 root   root   26 Jul  6 17:02 a_file_I_made_inside_a_container.txt
-rw-rw-r-- 1 surchs surchs 29 Jul  6 16:47 this_is_a_file_on_my_computer.txt
[surchs@marvin docker_things]$
```

path on host

path in container

file on host

https://docs.docker.com/storage/volumes/

How do I make my own Docker image?

1. Start **from** an existing image
2. Define **changes** on top of that with a **Dockerfile**
3. Build a new image using the **docker build** command
4. Use your new image and share it via an image registry (e.g. Dockerhub)

```
1 FROM continuumio/miniconda3:4.10.3-alpine
2
3 RUN conda install pytest -y
4
"~/Documents/docker_things/Dockerfile" 4 lines --100%--
```

```
[surchs@marvin docker_things]$ docker build -f Dockerfile -t my_test_image .
Sending build context to Docker daemon   16.9kB
Step 1/2 : FROM continuumio/miniconda3:4.10.3-alpine
 ---> 8afb5f84671e
Step 2/2 : RUN conda install pytest -y
 ---> Running in 4bbaa62be6e8
Collecting package metadata (current_repodata.json): ...working... done
Solving environment: ...working... done
```

https://docs.docker.com/engine/reference/builder/

There are tools to help make Dockerfiles

Welcome to Neurodocker!

Neurodocker is a command-line program that generates custom Dockerfiles and Singularity recipes for neuroimaging and minifies existing containers. Its purpose is to make it easier for scientists (and others) to easily create reproducible computational environments.

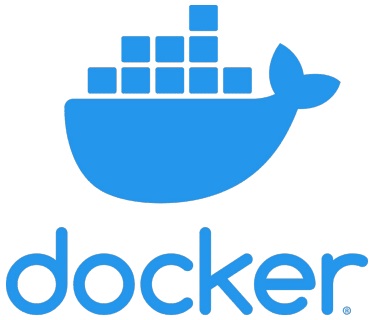
(This requires having Docker installed)

```
neurodocker generate docker --pkg-manager apt \
    --base-image neurodebian:buster \
    --ants version=2.3.4 \
    --miniconda version=latest conda_install="nipype notebook" \
    --user nonroot
```

https://www.repronim.org/neurodocker/user_guide/index.html

Docker: key points so far

- Docker **images** are **read-only snapshots**, you can find them on Dockerhub
- You can pull an image to run a container
- Images have tags (or version), that you should specify when pulling
- A Docker **container** is an isolated, **live instance** of an image
- Docker **containers** have their own file system, but this is **not persistent**
- With **volumes** we can **expose directories** on the host to the container
- We can **build** our own images on top of existing ones using a **Dockerfile**



Docker engine

```
# our base image
FROM alpine:3.5

# Install python and pip
RUN apk add --update py2-pip

# upgrade pip
RUN pip install --upgrade pip

# install Python modules needed by the Python app
COPY requirements.txt /usr/src/app/
```

Dockerfile

Docker image registry

OK, is that all?

On shared systems (like a supercomputer), you shouldn't / can't use Docker

- Docker isn't as isolated as a VM
- By default you run docker with root privileges and are root inside a container
- A malicious actor can escalate privileges and "break out" of a container

Apptainer (formerly Singularity) **is a container solution in these cases**

Singularity[1] is open source software created by Berkeley Lab:

- as a **secure way** to use Linux containers on Linux multi-user clusters,
- as a way to enable users to have **full control of their environment,** and,
- as a way to **package scientific software** and deploy such to *different* clusters having the *same* architecture.

i.e., it provides **operating-system-level virtualization** commonly called *containers*.

https://apptainer.org/docs/user/main/ and https://apptainer.org/docs/user/main/security.html

https://docs.alliancecan.ca/wiki/Singularity

Build images with Docker, run them with Singularity

1. Pull an image from Dockerhub and create a local **S**ingularity**I**mage**F**ile

```
$ apptainer pull docker://sylabsio/lolcow
```

2. Run the Singularity File using **apptainer run**

```
$ apptainer run lolcow_latest.sif
 _______________________________
< Mon Aug 16 13:01:55 CDT 2021 >
 -------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```

https://apptainer.org/docs/user/main/quick_start.html#

Overall recap

- use **Python virtual environments** (or conda) for Python projects
- isolate system level dependencies with **containers**
- **reuse** existing container images or **build your own** on top of others
- use **Apptainer** (Singularity) to run containers on **shared systems** (HPC)

Additional Resources

- Docker tutorial https://github.com/docker/labs
- Neurohackweek container course https://neurohackweek.github.io/docker-for-scientists/
- The Turing Way on reproducible research environments https://the-turing-way.netlify.app/reproducible-research/renv.html