ORIGAMI Lab

Containers

QLS 612 - May 2024

Sebastian Urchs - Alyssa Dai

Fondation Brain Canada Foundation

NIH

NIBIB

HEALTHY BRAINS HEALTHY LIVES

neuro Montreal Neurological Institute-Hospital

CONP PCNO

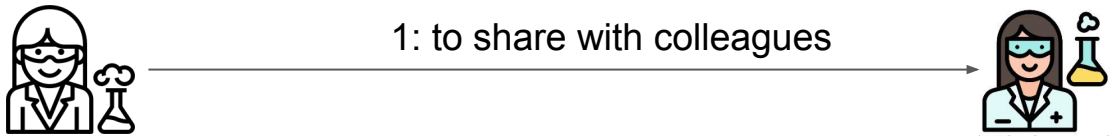
Containers?

What we will talk about

1. **Why** are containers useful for researchers
2. Using and building containers with **Docker**
3. Using containers on supercomputers with **Singularity**

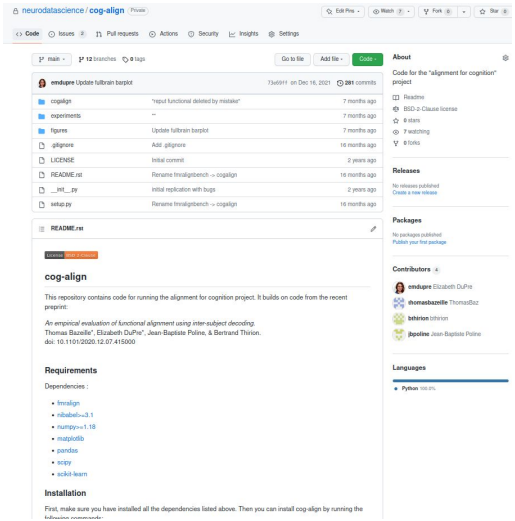
Why isolate environments

Document your software environment



Document your software environment

1: to share with colleagues



Requirements

Dependencies :

- fmralign
- nibabel>=3.1
- numpy>=1.18
- matplotlib
- pandas
- scipy
- scikit-learn

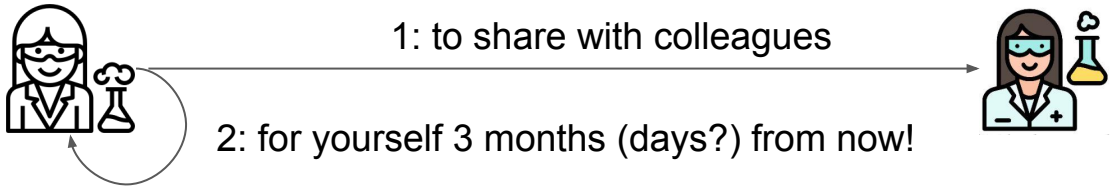
Installation

First, make sure you have installed all the dependencies listed above. Then you can install cog-align by running the following commands:

```
git clone https://github.com/neurodatascience/cog-align
cd cog-align
pip install -e .
```

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

Document your software environment




```
Python 2.7.18

>>> print "hello"
hello
```

What's New In Python 3.0

Author: Guido van Rossum

Print Is A Function

The `print` statement has been replaced with a `print()` function, with keyword arguments to replace most of the special syntax of the old `print` statement (PEP 3105). Examples:

```
Old: print "The answer is", 2*2
New: print("The answer is", 2*2)

Old: print x,           # Trailing comma suppresses newline
New: print(x, end=" ")  # Appends a space instead of a newline

Old: print              # Prints a newline
New: print()            # You must call the function!
```

```
Python 3.11.7

>>> print "hello"
  File "<stdin>", line 1
    print "hello"
    ^^^^^^^^^^^^
SyntaxError: Missing parentheses in call to 'print'. Did you mean print( ... )?
```

Document your software environment

1: to share with colleagues

2: for yourself 3 months (days?) from now!

```
Python 2.7.18

>>> print "hello"
hello
```

What's New In Python 3.0

Author: Guido van Rossum

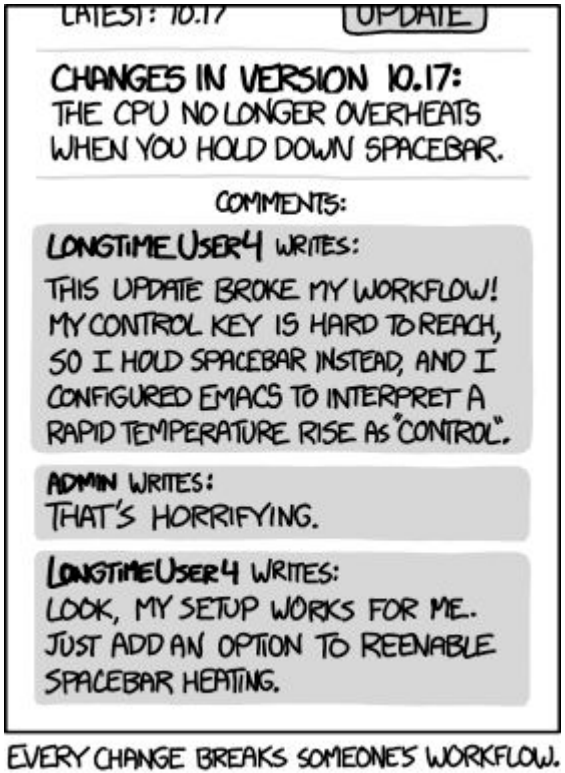
Print Is A Function

The `print` statement has been replaced with a `print()` function, with keyword arguments to replace most of the special syntax of the old `print` statement (PEP 3105). Examples:

```
Old: print "The answer is", 2*2
New: print("The answer is", 2*2)

Old: print x,           # Trailing comma suppresses newline
New: print(x, end=" ")  # Appends a space instead of a newline

Old: print              # Prints a newline
New: print()            # You must call the function!
```

```
Python 3.11.7

>>> print "hello"
  File "<stdin>", line 1
    print "hello"
    ^^^^^^^^^^^^
SyntaxError: Missing parentheses in call to 'print'. Did you mean print( ... )?
```

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

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`.

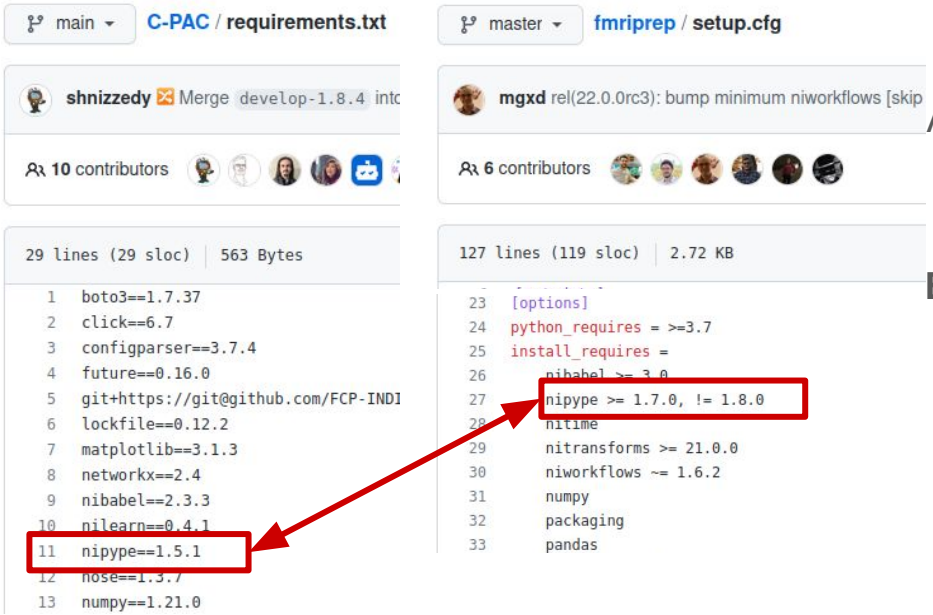
On such systems, it is often better to use a virtual environment or a per-user installation when installing packages with `pip`.

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

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

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**

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! 🙂

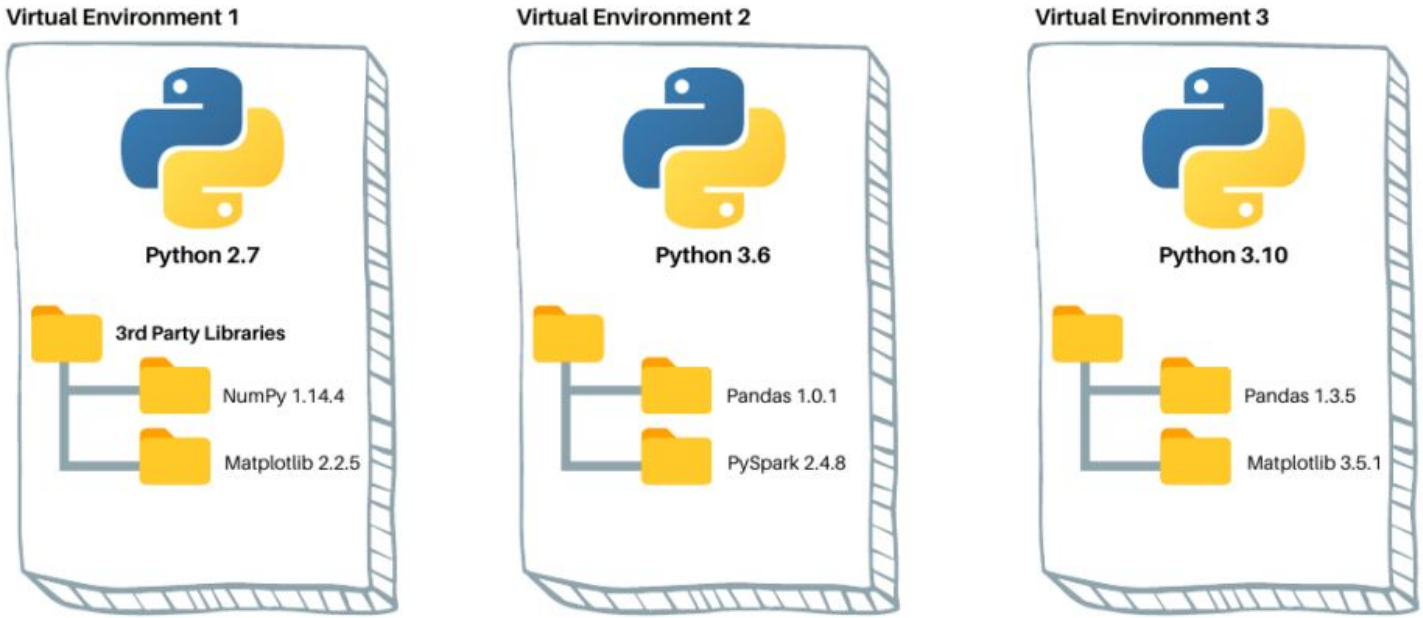
5 **Bake:** Bake the cream cheese pound cake at 325°F (163°C). Hal
the cake with aluminum foil to prevent over-browning.

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

isolate Python environments



Why not just python virtual environments?

External Dependencies

fMRIPrep is written using Python 3.8 (or above), and is based on nipype.

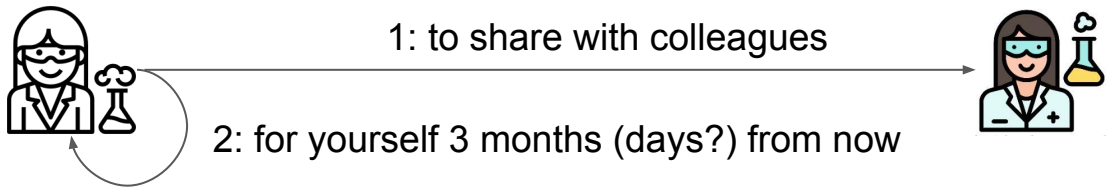
fMRIPrep requires some other neuroimaging software tools that are not handled by the Python's packaging system (Pypi) used to deploy the `fmriprep` package:

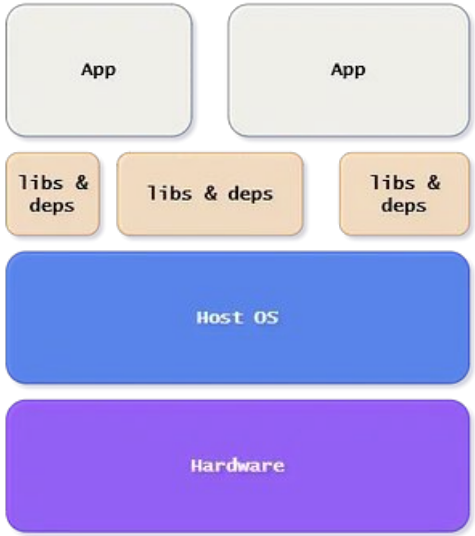
- FSL (version 6.0.5.1)
- ANTs (version 2.3.3 - NeuroDocker build)
- AFNI (version 22.3.06)
- C3D (version 1.3.0)
- FreeSurfer (version 7.3.2)
- ICA-AROMA (version 0.4.5)
- bids-validator (version 1.8.0)
- connectome-workbench (version 1.5.0)

https://fmriprep.org/en/stable/installation.html

- 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

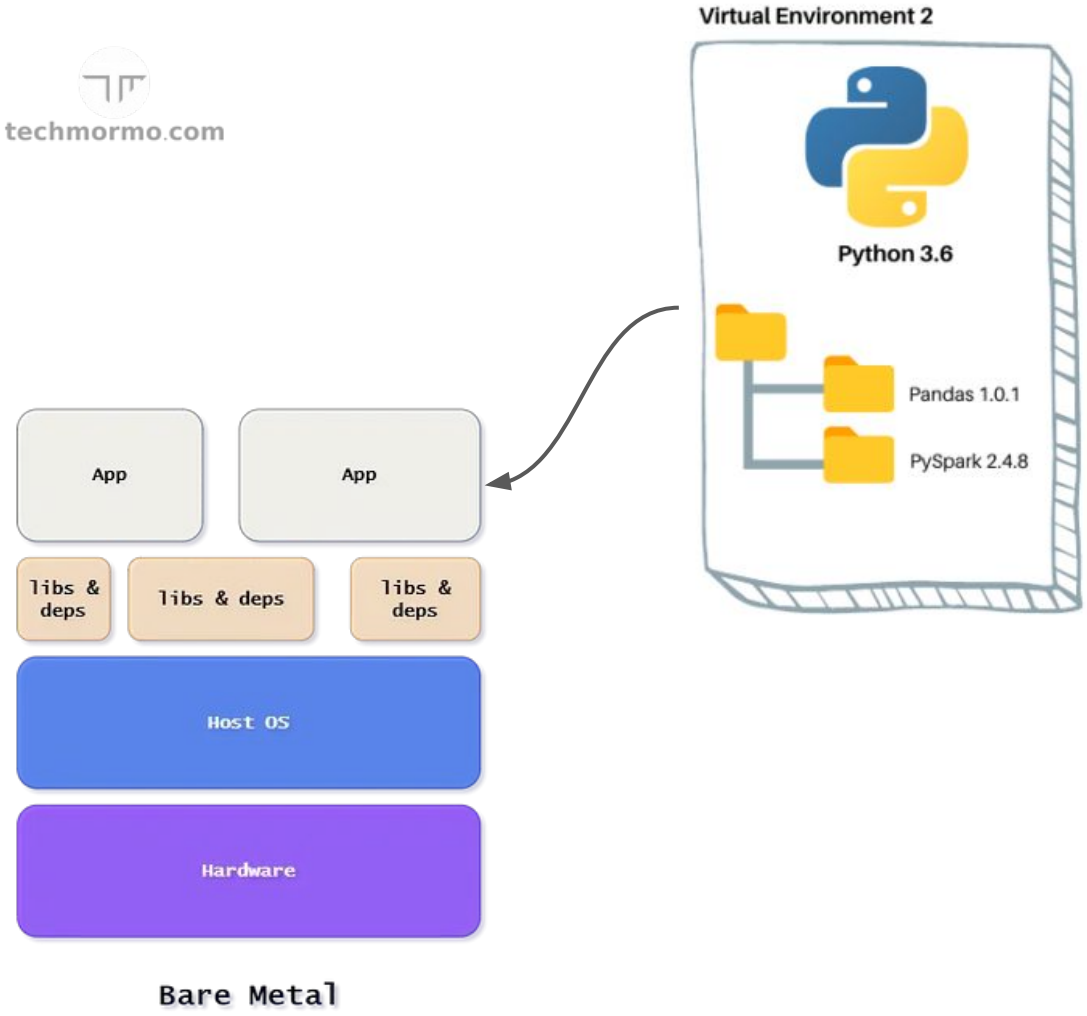
Need to capture the (entire) compute environment



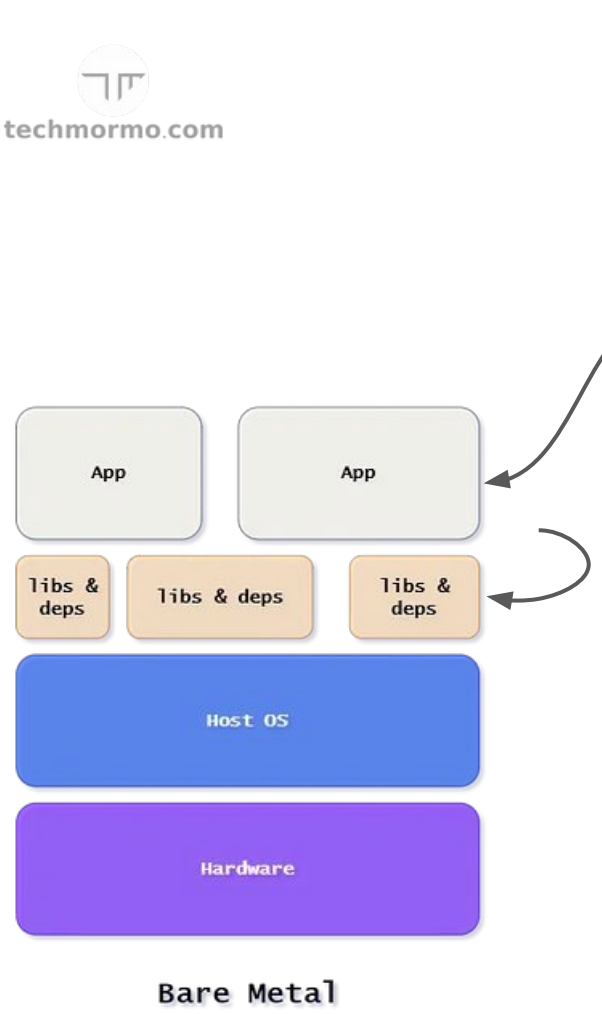


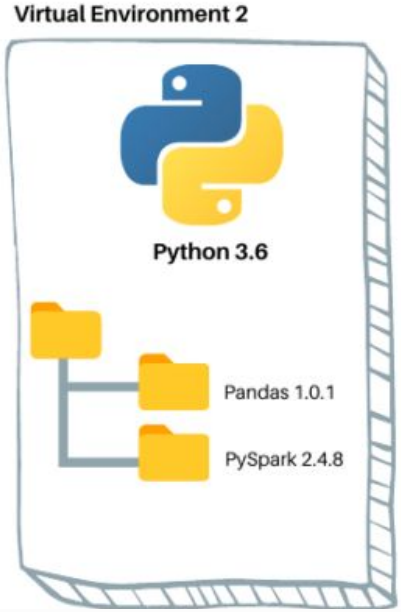
Bare Metal

techmormo.com



techmormo.com





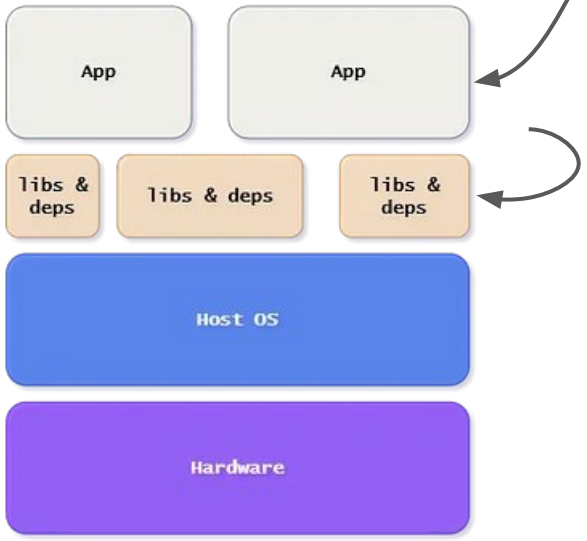
External Dependencies

fMRIPrep is written using Python 3.8 (or above), and is based on nipype.

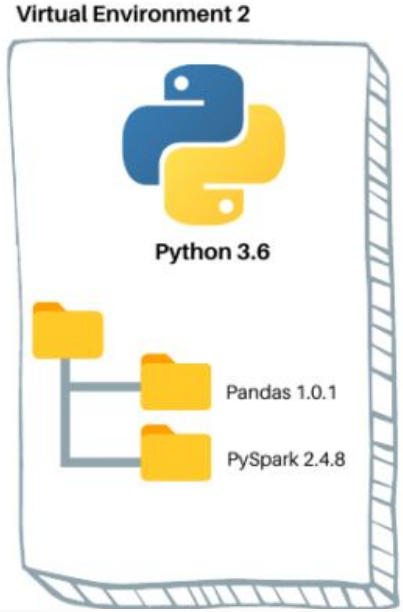
fMRIPrep requires some other neuroimaging software tools that are not handled by the Python's packaging system (Pypi) used to deploy the `fmriprep` package:

- FSL (version 6.0.5.1)
- ANTs (version 2.3.3 - NeuroDocker build)
- AFNI (version 22.3.06)
- C3D (version 1.3.0)
- FreeSurfer (version 7.3.2)
- ICA-AROMA (version 0.4.5)
- bids-validator (version 1.8.0)
- connectome-workbench (version 1.5.0)

techmormo.com



Bare Metal



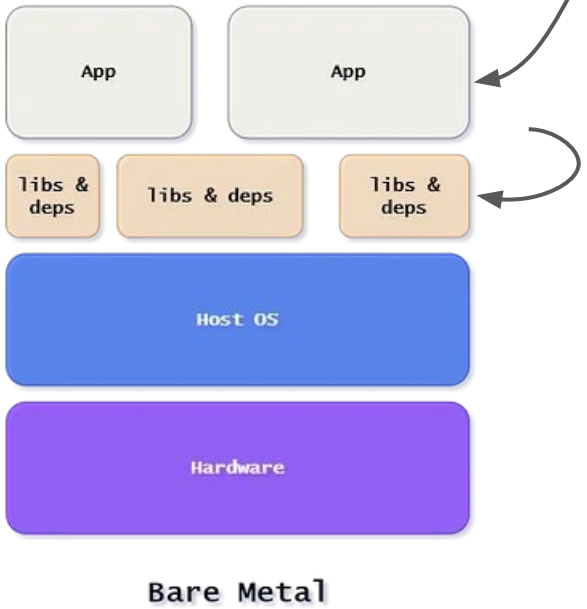
External Dependencies

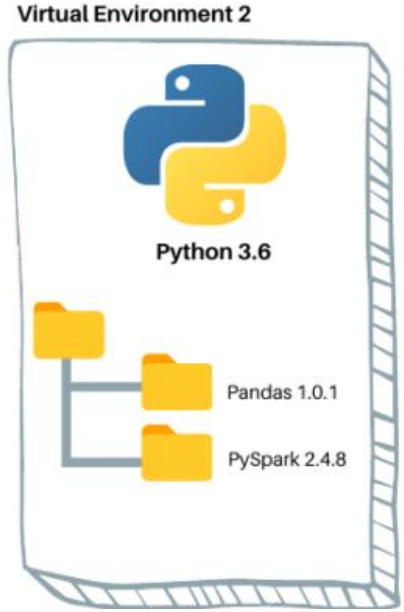
fMRIPrep is written using Python 3.8 (or above), and is based on nipype.

fMRIPrep requires some other neuroimaging software tools that are not handled by the Python's packaging system (Pypi) used to deploy the `fmriprep` package:

- FSL (version 6.0.5.1)
- ANTs (version 2.3.3 - NeuroDocker build)
- AFNI (version 22.3.06)
- C3D (version 1.3.0)
- FreeSurfer (version 7.3.2)
- ICA-AROMA (version 0.4.5)
- bids-validator (version 1.8.0)
- connectome-workbench (version 1.5.0)

techmormo.com





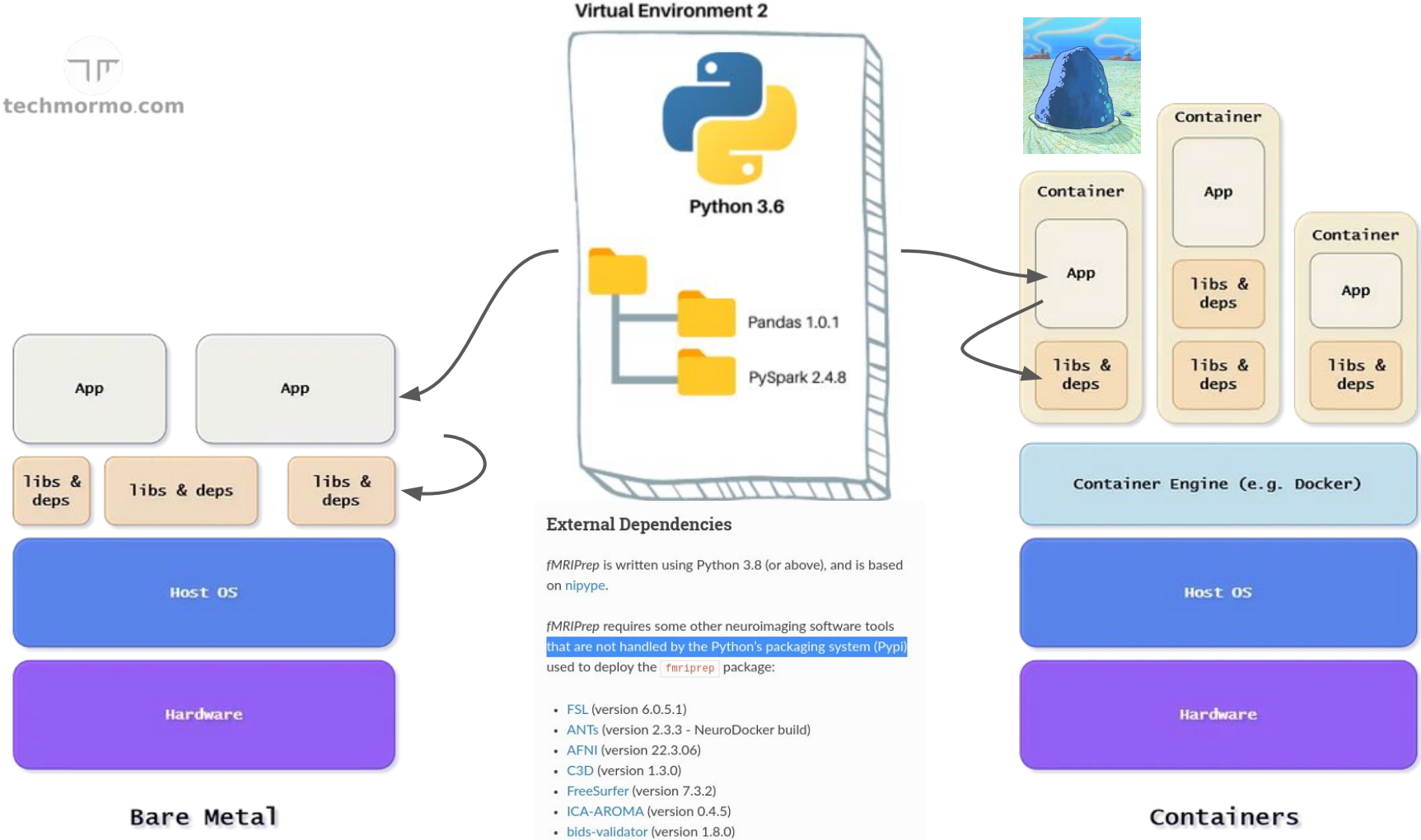
External Dependencies

fMRIPrep is written using Python 3.8 (or above), and is based on nipype.

fMRIPrep requires some other neuroimaging software tools that are not handled by the Python's packaging system (Pypi) used to deploy the `fmriprep` package:

- FSL (version 6.0.5.1)
- ANTs (version 2.3.3 - NeuroDocker build)
- AFNI (version 22.3.06)
- C3D (version 1.3.0)
- FreeSurfer (version 7.3.2)
- ICA-AROMA (version 0.4.5)
- bids-validator (version 1.8.0)
- connectome-workbench (version 1.5.0)

techmormo.com



External Dependencies

fMRIPrep is written using Python 3.8 (or above), and is based on nipype.

fMRIPrep requires some other neuroimaging software tools that are not handled by the Python's packaging system (Pypi) used to deploy the `fmriprep` package:

- FSL (version 6.0.5.1)
- ANTs (version 2.3.3 - NeuroDocker build)
- AFNI (version 22.3.06)
- C3D (version 1.3.0)
- FreeSurfer (version 7.3.2)
- ICA-AROMA (version 0.4.5)
- bids-validator (version 1.8.0)
- connectome-workbench (version 1.5.0)

Containers

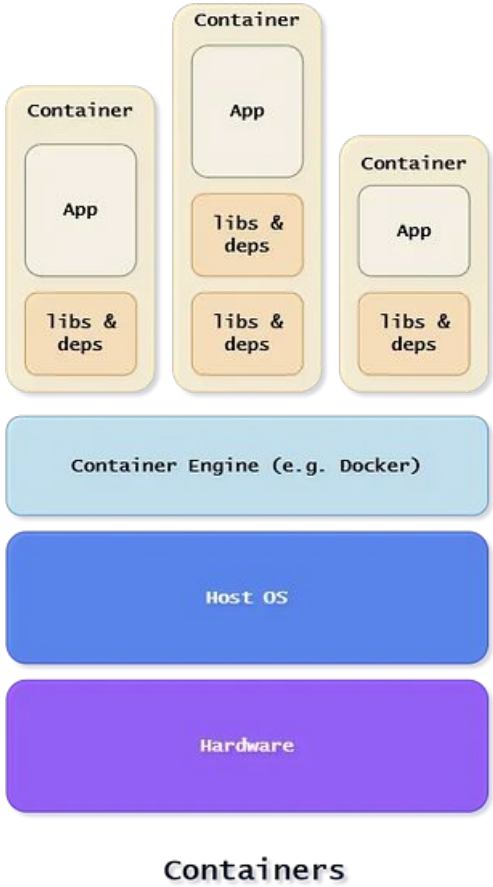
Containers

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

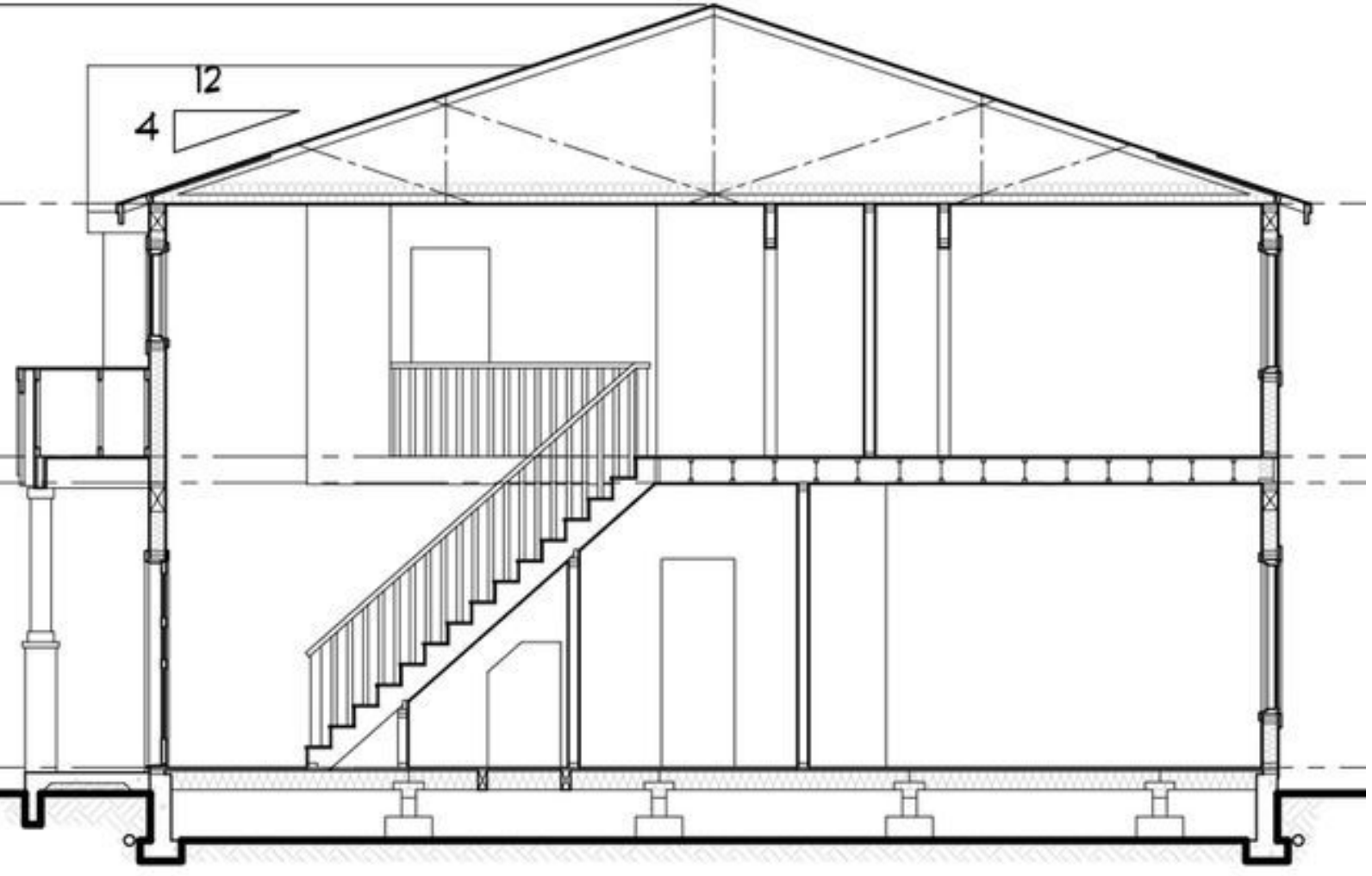
How do I make one

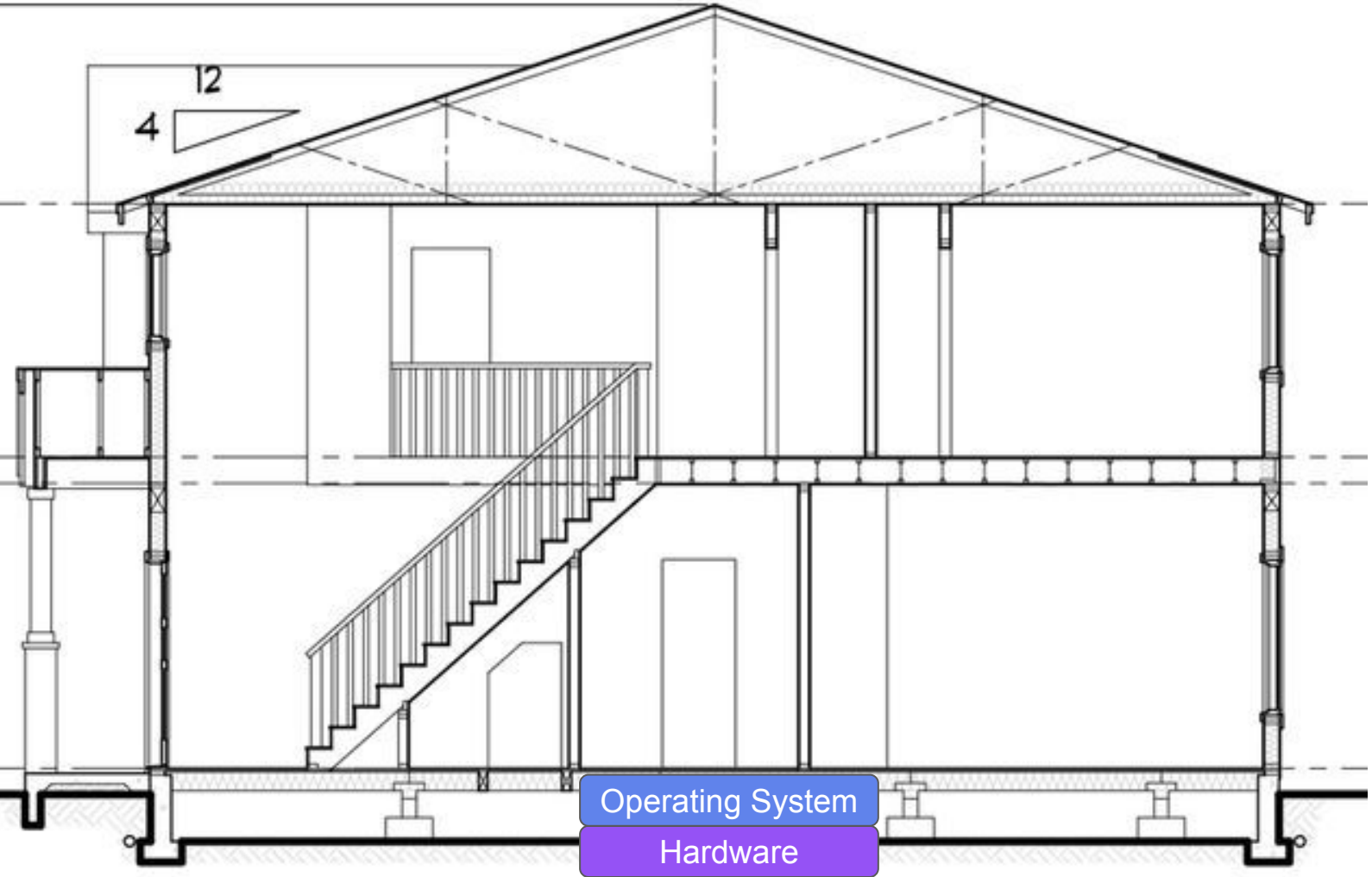
- use a container implementation
- **docker** is the most widely used
- Singularity is used on supercomputers

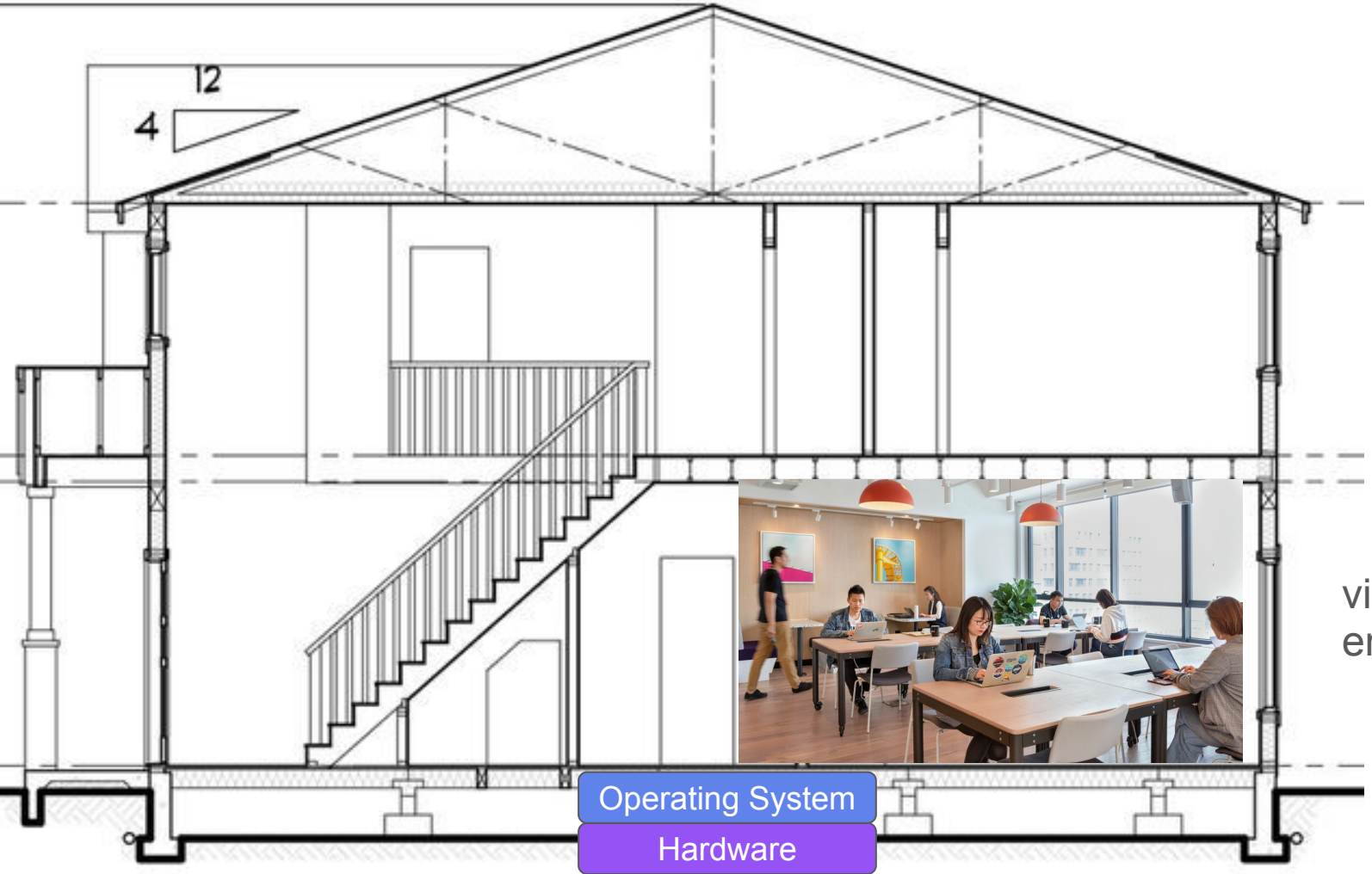


Containers

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


12
4

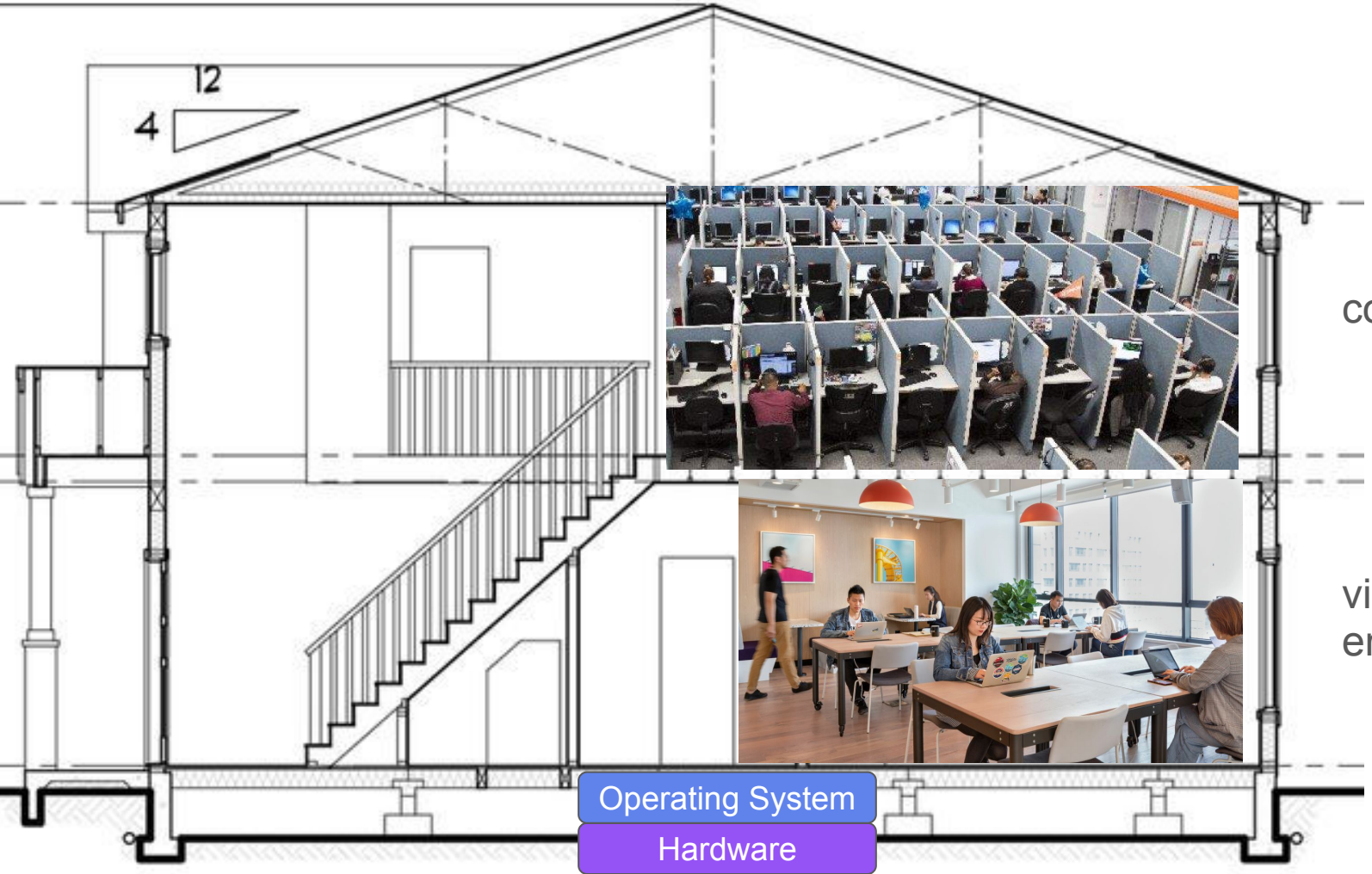

12
4
Operating System
Hardware



Operating System

Hardware

virtual environments

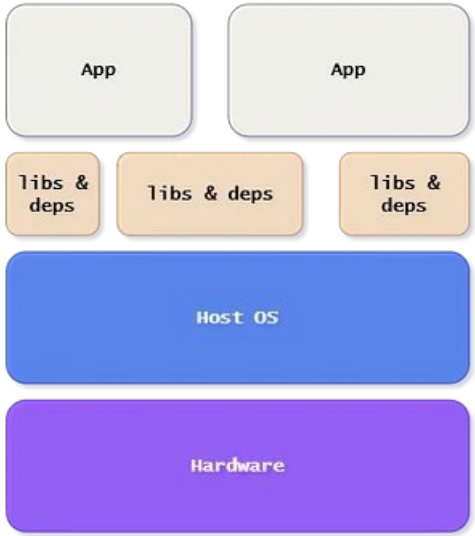


containers

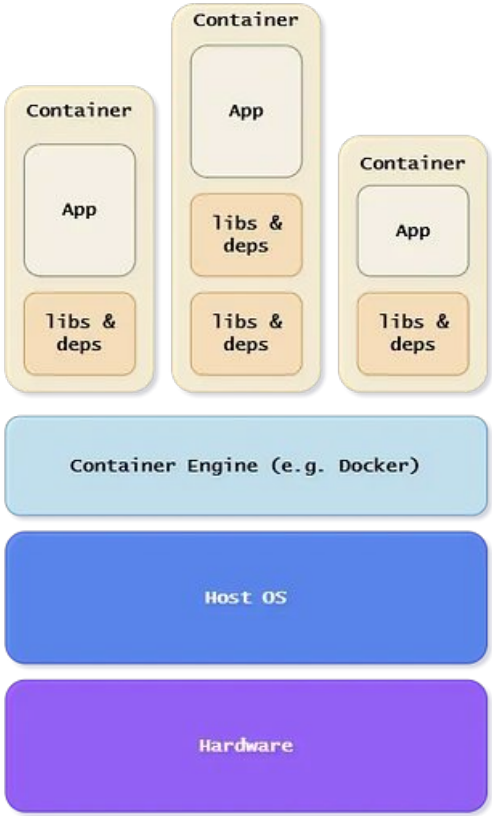
virtual environments

Operating System

Hardware

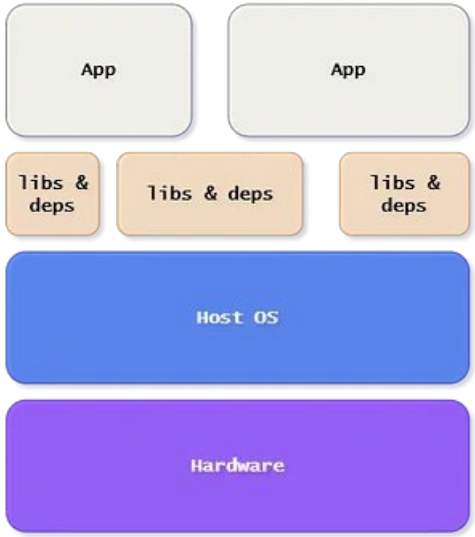


Bare Metal

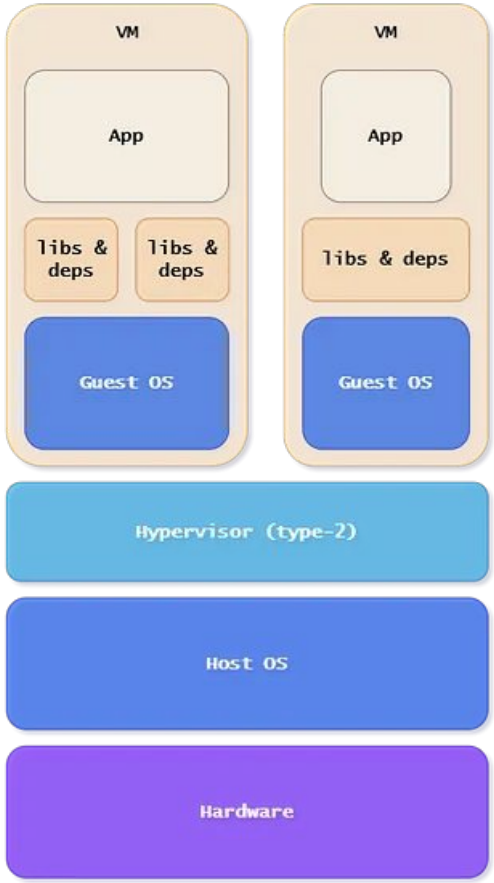


Containers

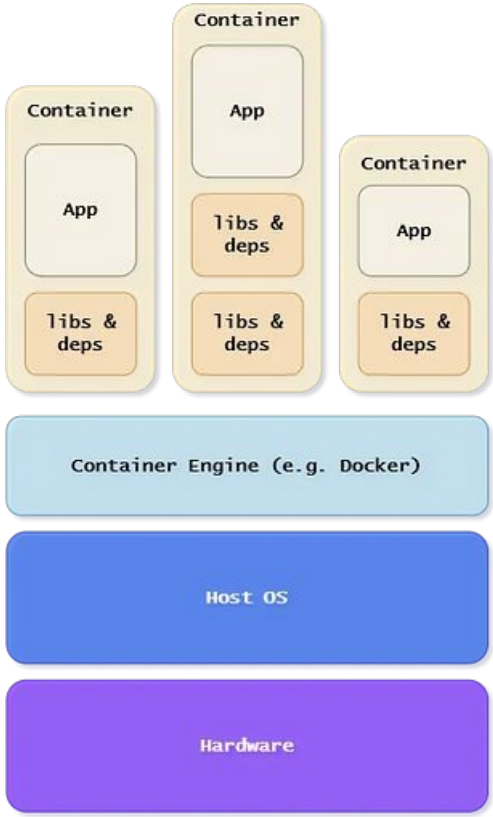
techmormo.com



Bare Metal

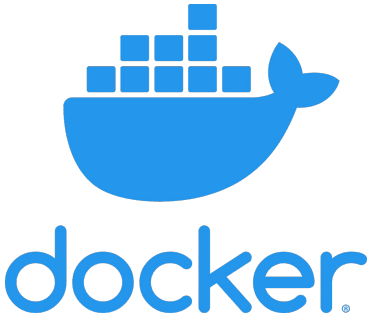


Virtual Machines



Containers

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/

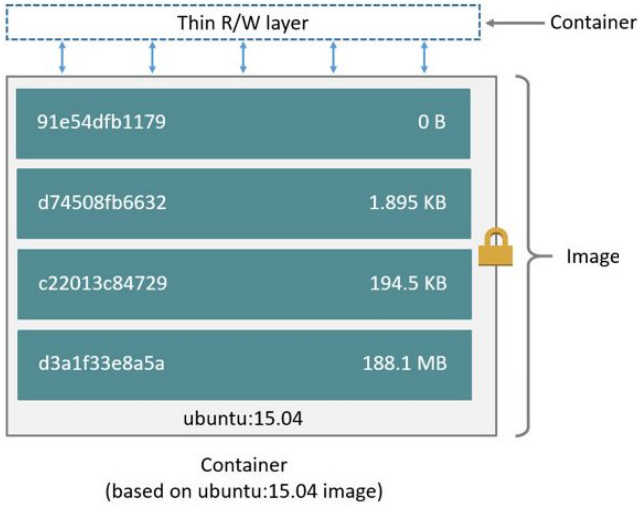
Docker **image** and **container**: what's the difference

Docker image

- a **read-only** snapshot of an environment
- 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**

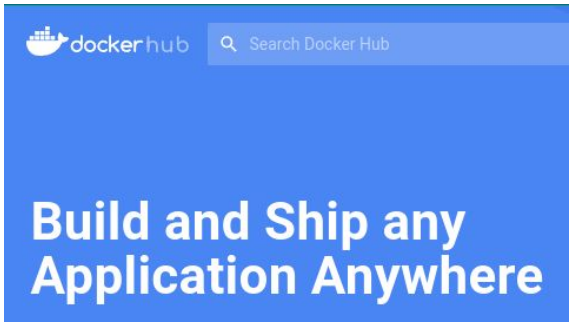


https://docs.docker.com/storage/storagedriver/

How can I get my own Docker container going

Use an **existing image**

- Dockerhub: a repository of docker images https://hub.docker.com/
- You can pull an image with `docker pull`

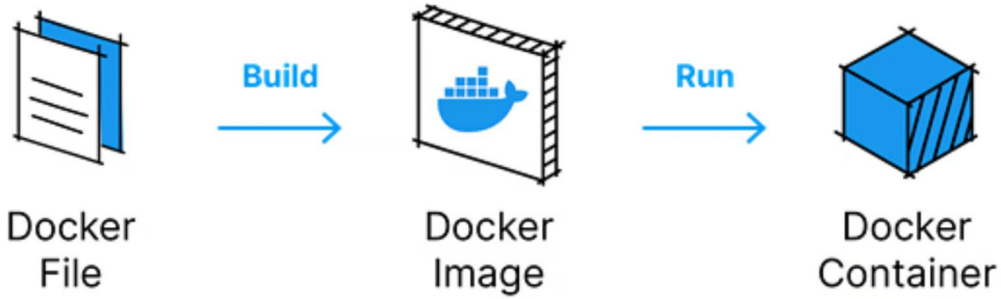


Build your **own image**

- a `Dockerfile` lets you describe the exact image you want to create
- Start from one image and add to it

```
# syntax=docker/dockerfile:1
FROM python:3.10-slim-buster
WORKDIR /app
COPY . /app/src
RUN pip install -r /app/src/requirements.txt
RUN pip install --no-cache-dir --no-deps /app/src[all]

ENTRYPOINT [ "bagel" ]
CMD ["--help"]
```

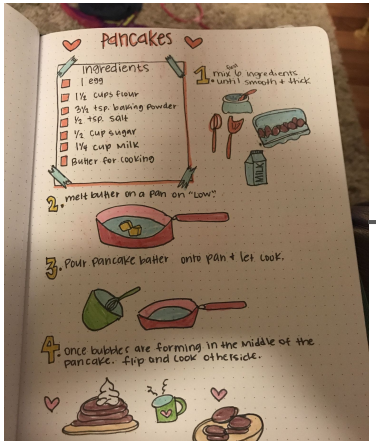
Let's look at both


Build
Run
Docker File
Docker Image
Docker Container

Docker File

Docker Image

Docker Container



recipe

combine

dough

make from

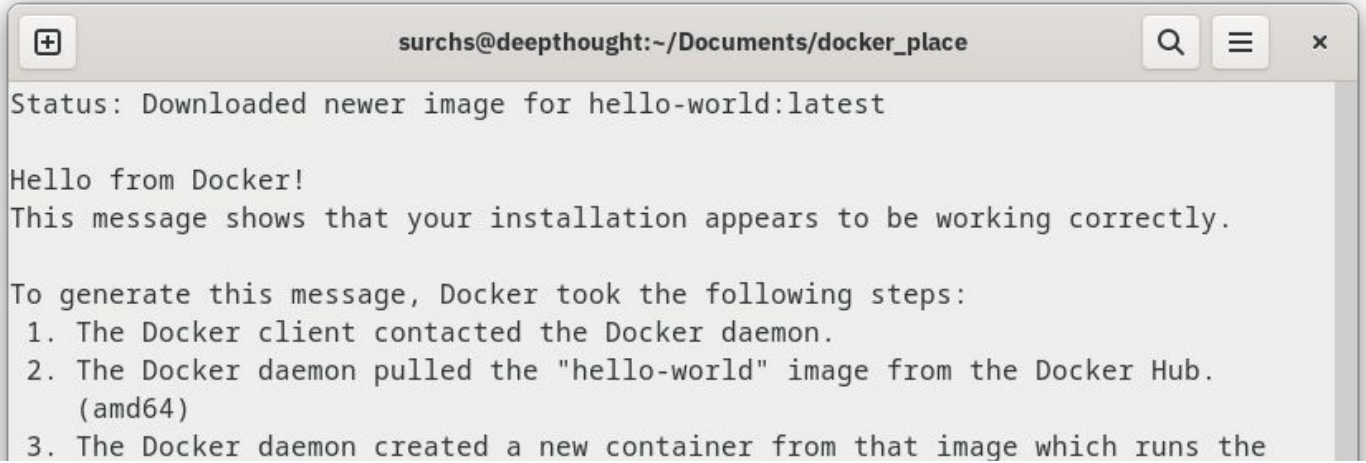
pancake (with stuff)

Exercise 1: Run a container from Dockerhub

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

Exercise 1: Run a container from Dockerhub

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



Exercise 1: Run a container from Dockerhub

Summary

- Dockerhub has images
- Image tags are important
- We can
  - retrieve images with `docker pull`
  - create and immediately run a container from an image with `docker run`

Let’s find a more useful image

Exercise 2: work with a container that has conda

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/
- Pick a tag, then pull the image `$ docker pull continuumio/miniconda3:22.11.1-alpine`
- Run it

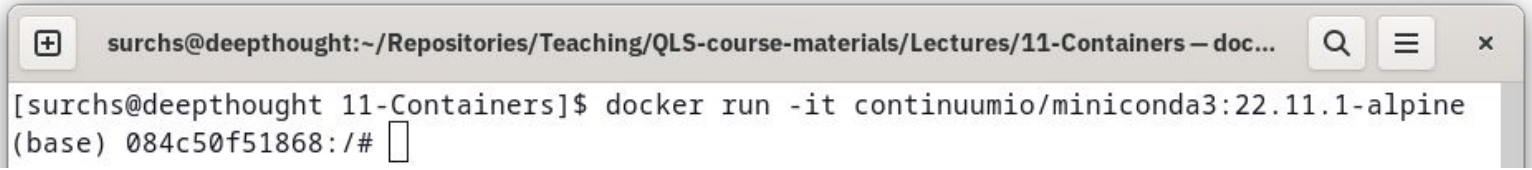
Exercise 2: work with a container that has conda

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/
- Pick a tag, then pull the image `$ docker pull continuumio/miniconda3:22.11.1-alpine`
- Run it

Oh, nothing happened?

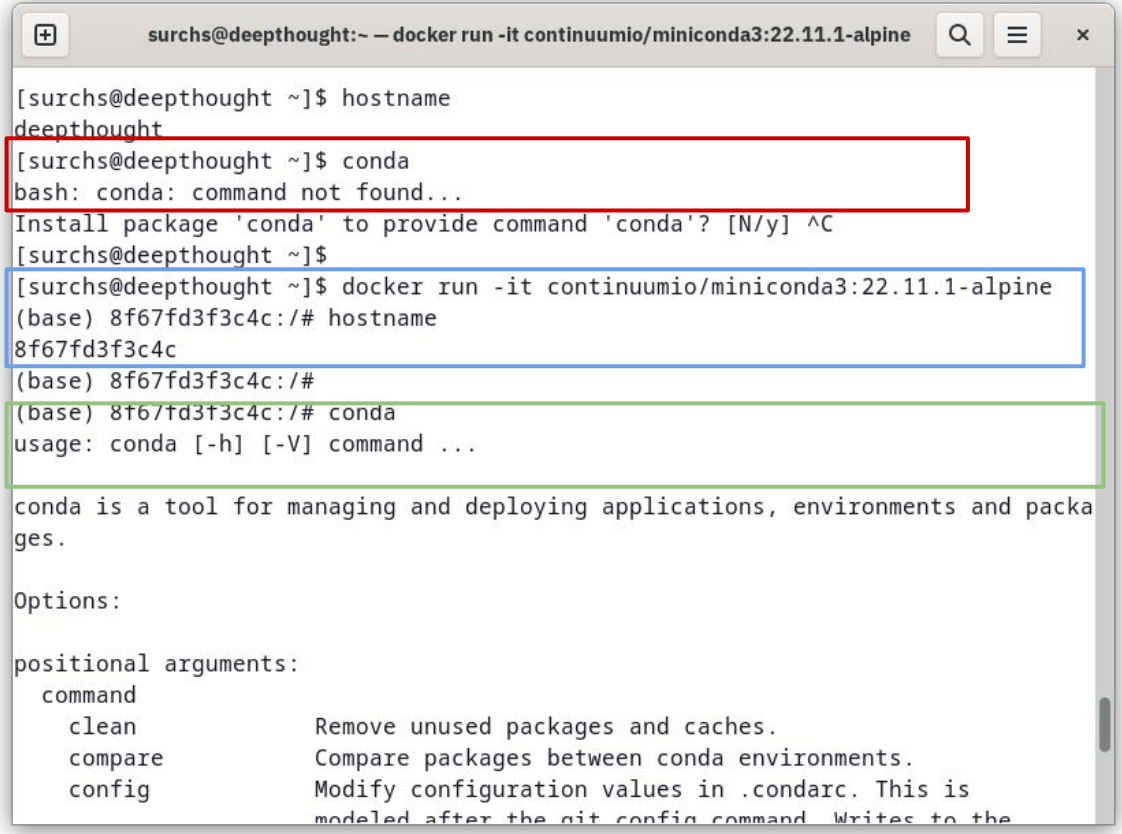
- Let’s run it interactively to take a look inside



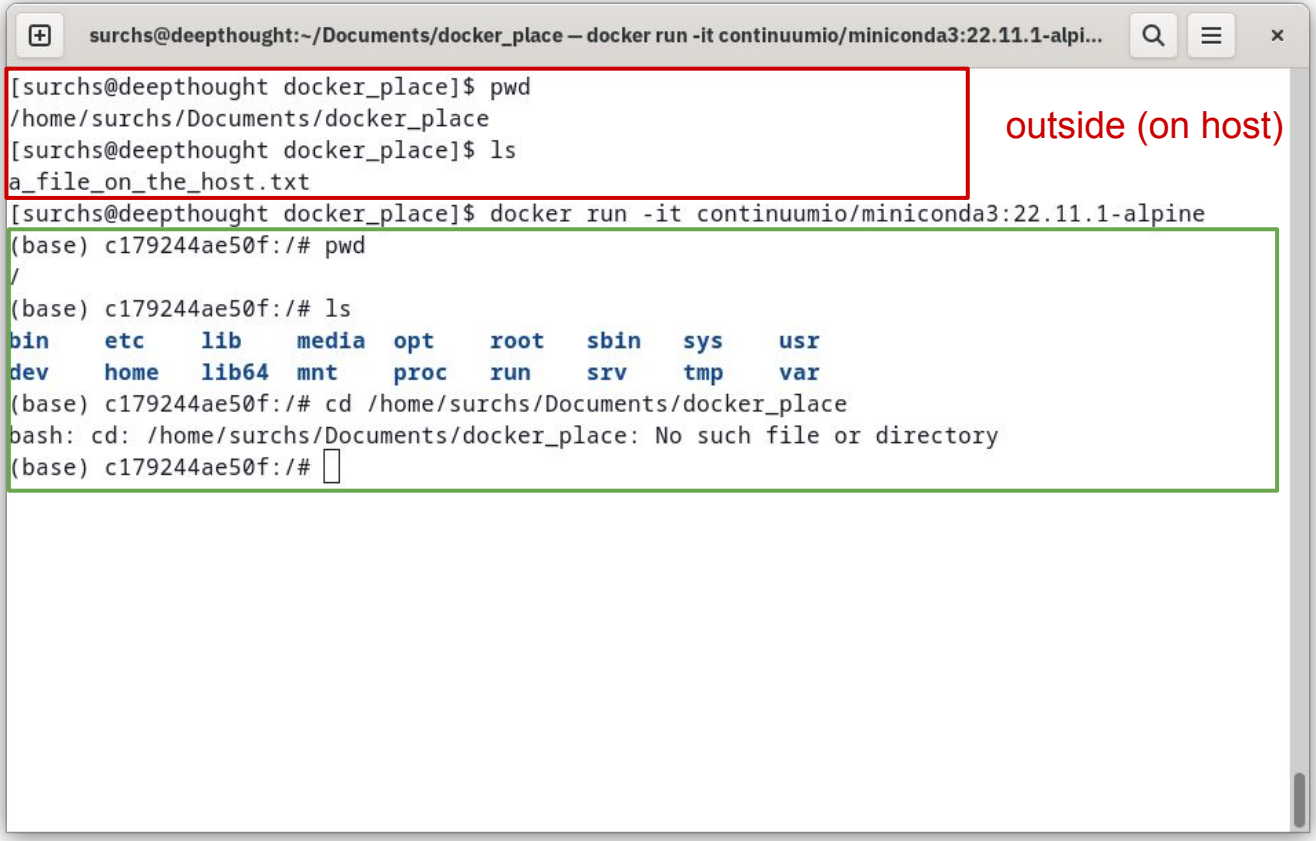
https://github.com/ContinuumIO/docker-images/blob/master/miniconda3/alpine/Dockerfile

Looking around inside a container

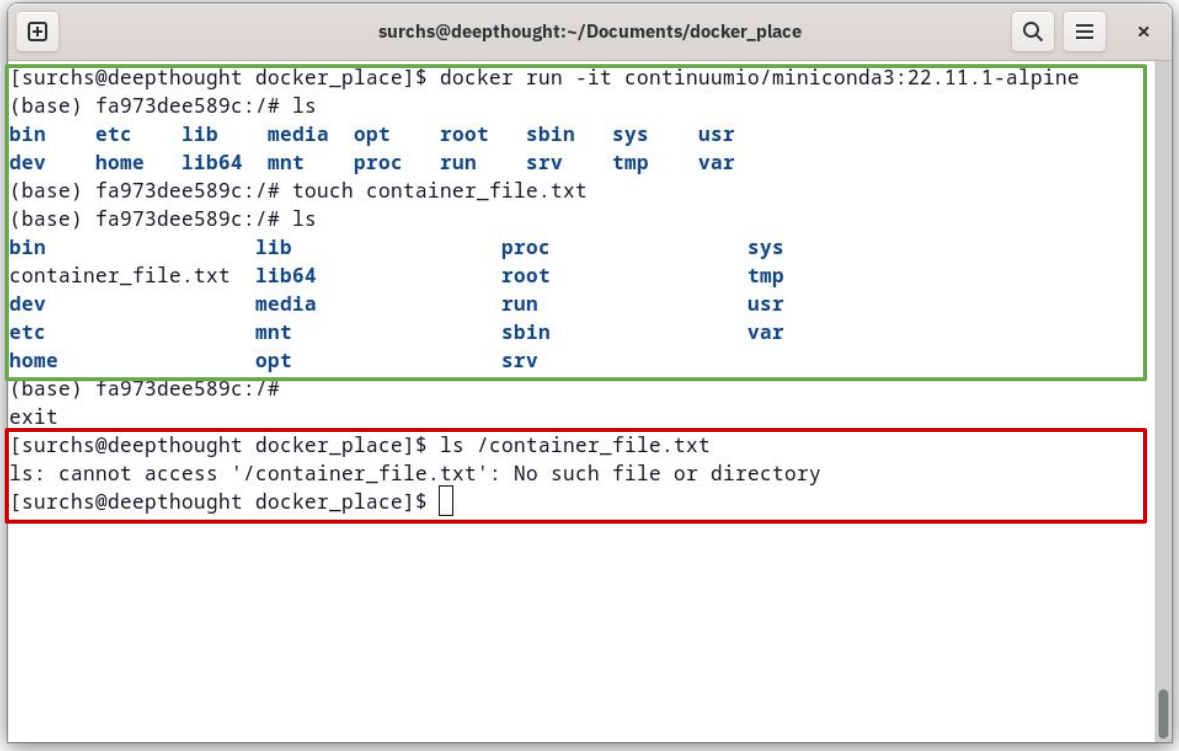
- no conda on my machine
- starting container changes the look of my terminal and the name of my computer
- inside of the container I have access to conda



By default the container doesn’t see files on the host …



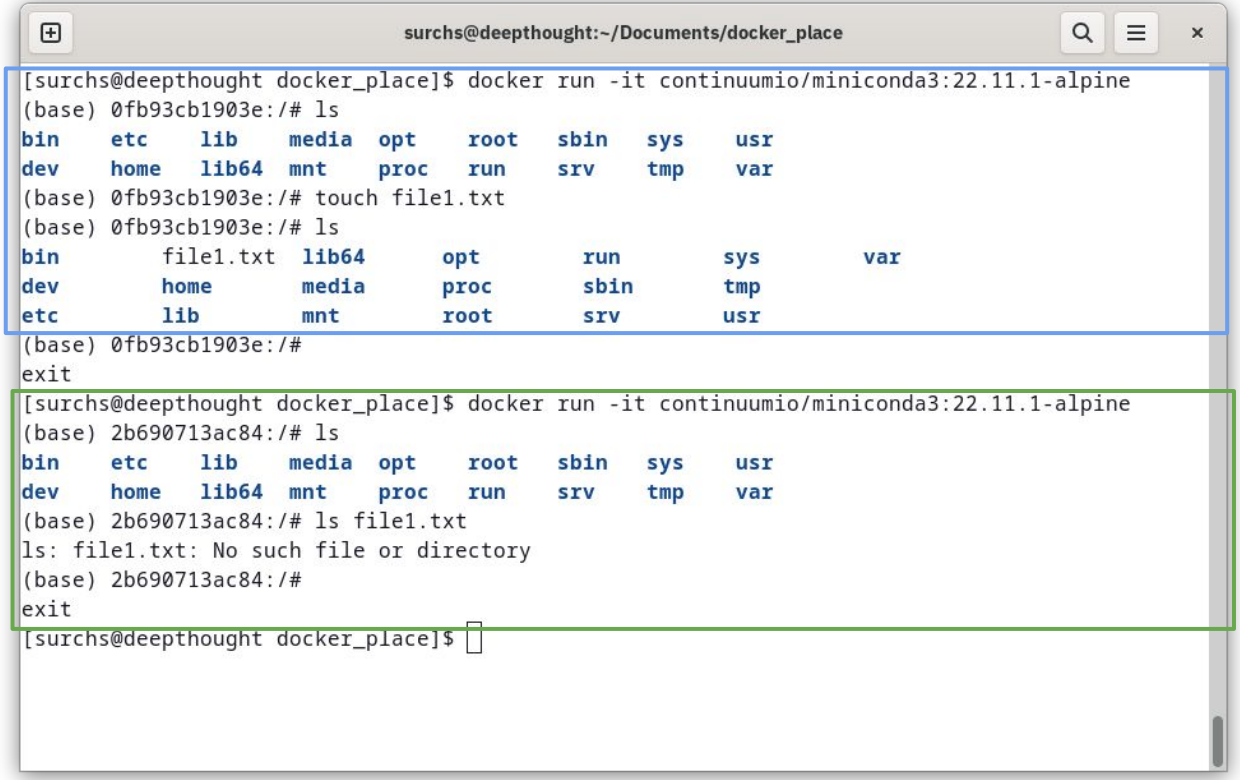
… and the host can’t see files on the container …



inside
(in container)

outside (on host)

… and files written to a container are tied to it!



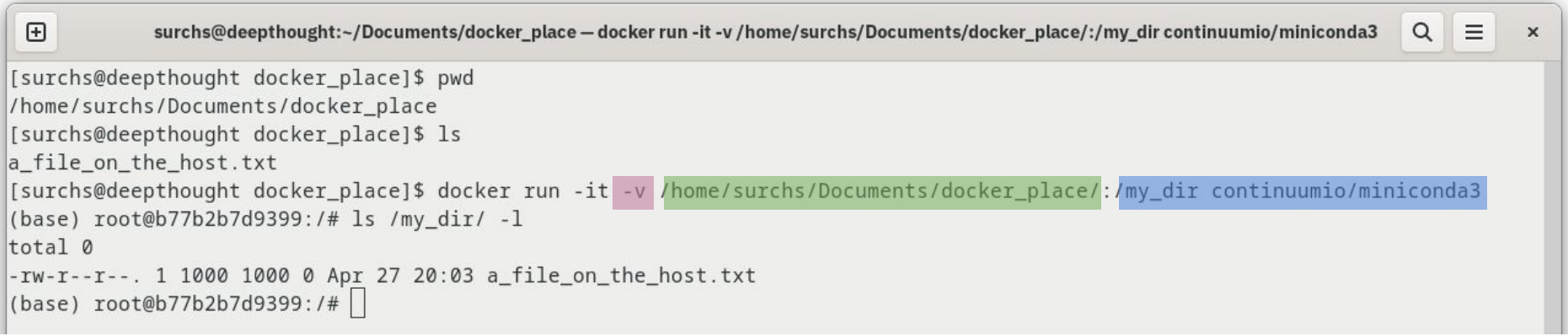
first container instance

second container instance

-> Don’t write anything (worth keeping) to a container

How do we share files between host and container?

- Mount a path on the host to a path in the container



- The bind-mount will be created in the container, even if it exists already!
- Make sure you provide a path (starts with / or ./) - or use `--mount`

https://docs.docker.com/storage/bind-mounts/

Exercise 2: work with a container that has conda

Summary

- We can connect to an interactive shell in a container with `docker run -it`
- By default the container cannot see or write the filesystem of the host
- We can “mount” a path on the host into the container with

```
docker run -v /host/path:/container/path OR
docker run --mount type=bind,source=/host/path,target=/container/path
```

- It’s a bad idea to write into the container directly

So how do I make persistent changes to the image?

Exercise 3: Run a Dockerized tool on local data

We've explored how a Docker container behaves on the inside. Let's follow the more "typical" steps to run a tool available as a Docker image.

inside

outside

Exercise 3: Run a Dockerized tool on local data

Let's try running the BIDS validator on a test dataset `ds006`, to see if the dataset complies with the Brain Imaging Dataset Structure.

1. **Pull** the bids/validator Docker image tagged `latest`, and **list** the images on your system to confirm that bids/validator is there

```
docker pull bids/validator:latest
docker images
```

Exercise 3: Run a Dockerized tool on local data

Recall: `docker run [options-for-docker] <imagename> <command>`

Let's print the help text for the bids/validator:

```
docker run --rm bids/validator --help
```

Output:

```
Usage: bids-validator <dataset_directory> [options]
…
```

Exercise 3: Run a Dockerized tool on local data

Recall: `docker run [options-for-docker] <imagename> <command>`

Let's print the help text for the bids/validator:

```
docker run --rm bids/validator --help
```

Output:

```
Usage: bids-validator <dataset_directory> [options]
…
```

…How do I run other validator commands?

```
docker run --rm bids/validator <dataset_directory> [options]
```

Exercise 3: Run a Dockerized tool on local data

```
docker run --rm bids/validator <dataset_directory> [options]
```

Our test dataset is located **on our host machine** under:
`QLS-course-materials/Lectures/2024/data/ds006`

2. Run the bids/validator image again, but provide this dataset directory to validate.

HINTS:

- **Mount** the dataset directory into the container for it to find the dataset (remember: the path must start with / or ./ !)

Exercise 3: Run a Dockerized tool on local data

```
docker run --rm bids/validator <dataset_directory> [options]
```

Our test dataset is located **on our host machine** under:
`QLS-course-materials/Lectures/2024/data/ds006`

2. Run the bids/validator image again, but provide this dataset directory to validate.

HINTS:

- **Mount** the dataset directory into the container for it to find the dataset (remember: the path must start with / or ./ !)
- `-v /path/to/data/on/host:/path/to/mount/data/in/container`

Exercise 3: Run a Dockerized tool on local data

```
docker run --rm bids/validator <dataset_directory> [options]
```

Our test dataset is located **on our host machine** under:
`QLS-course-materials/Lectures/2024/data/ds006`

2. Run the bids/validator image again, but provide this dataset directory to validate.

HINTS:

- **Mount** the dataset directory into the container for it to find the dataset (remember: the path must start with / or ./ !)
- `-v /path/to/data/on/host:/path/to/mount/data/in/container`

(from `data/` directory)

```
docker run --rm -v ./ds006:/data bids/validator /data
```

Exercise 3: Run a Dockerized tool on local data

The BIDS validator returns errors `[ERR]` because the test dataset contains empty data files.

3. To make the validator happy, run the Docker image again on the dataset, but this time add **command arguments** to:
   - Ignore warnings
   - Ignore NIfTI header content

HINT:
   - Print the help text again to see all command options

Exercise 3: Run a Dockerized tool on local data

The BIDS validator returns errors `[ERR]` because the test dataset contains empty data files.

3. To make the validator happy, run the Docker image again on the dataset, but this time add **command arguments** to:
   - Ignore warnings
   - Ignore NIfTI header content

HINT:
   - Print the help text again to see all command options

```
docker run --rm -v ./ds006:/data bids/validator /data
--ignoreNiftiHeaders --ignoreWarnings
```

Apptainer (singularity)

Apptainer is the container solution for HPCs

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 Apptainer

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**

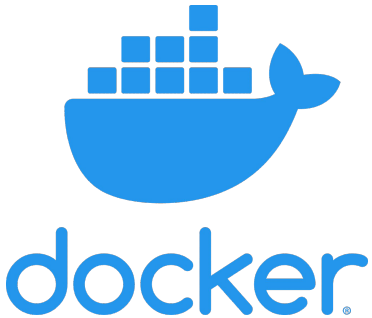
```
$ apptainer run lolcow_latest.sif
 _______________________________
< Mon Aug 16 13:01:55 CDT 2021 >
 -------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```

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

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

Container Summary

- Docker **images** are **read-only snapshots**, you can find them on Dockerhub
- 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**
- Use **Apptainer** to run Docker images on a compute cluster



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

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
```

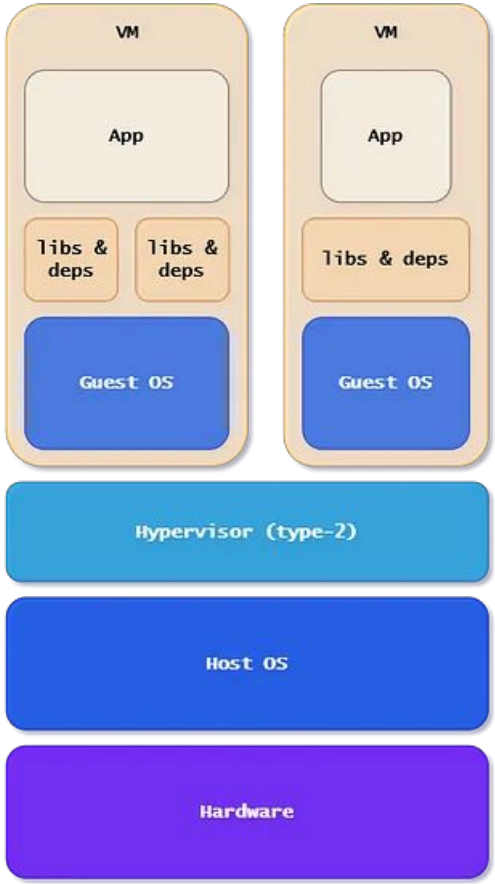
https://www.repronim.org/neurodocker/user_guide/index.html

Additional Resources

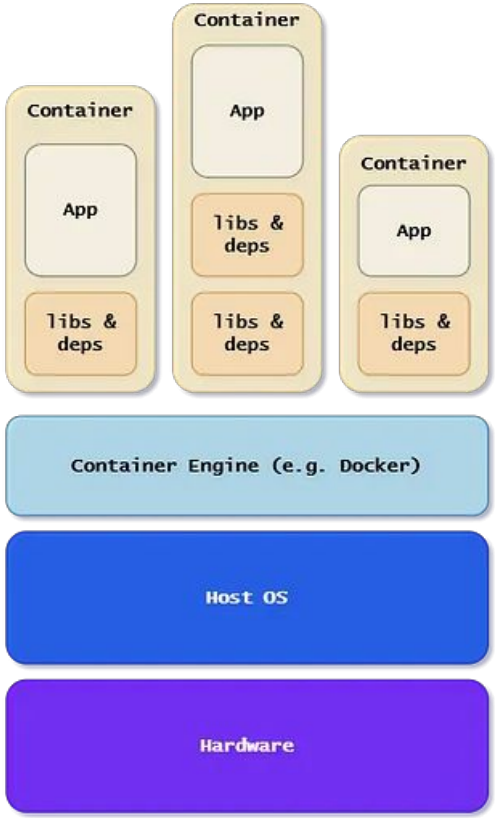
- Docker tutorial https://github.com/docker/labs
- Neurohackweek container course https://neurohackweek.github.io/docker-for-scientists/
- The Turing Way on reproducible research environments https://the-turing-way.netlify.app/reproducible-research/renv.html

techmormo.com

Bare Metal



Virtual Machines



Containers