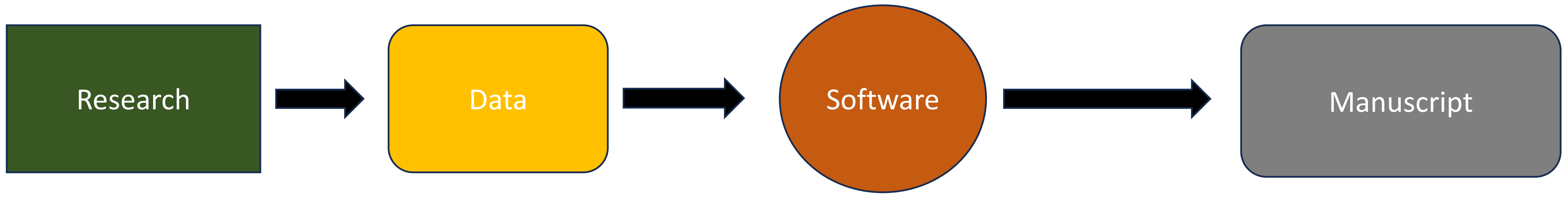
# Dependency management with Conda

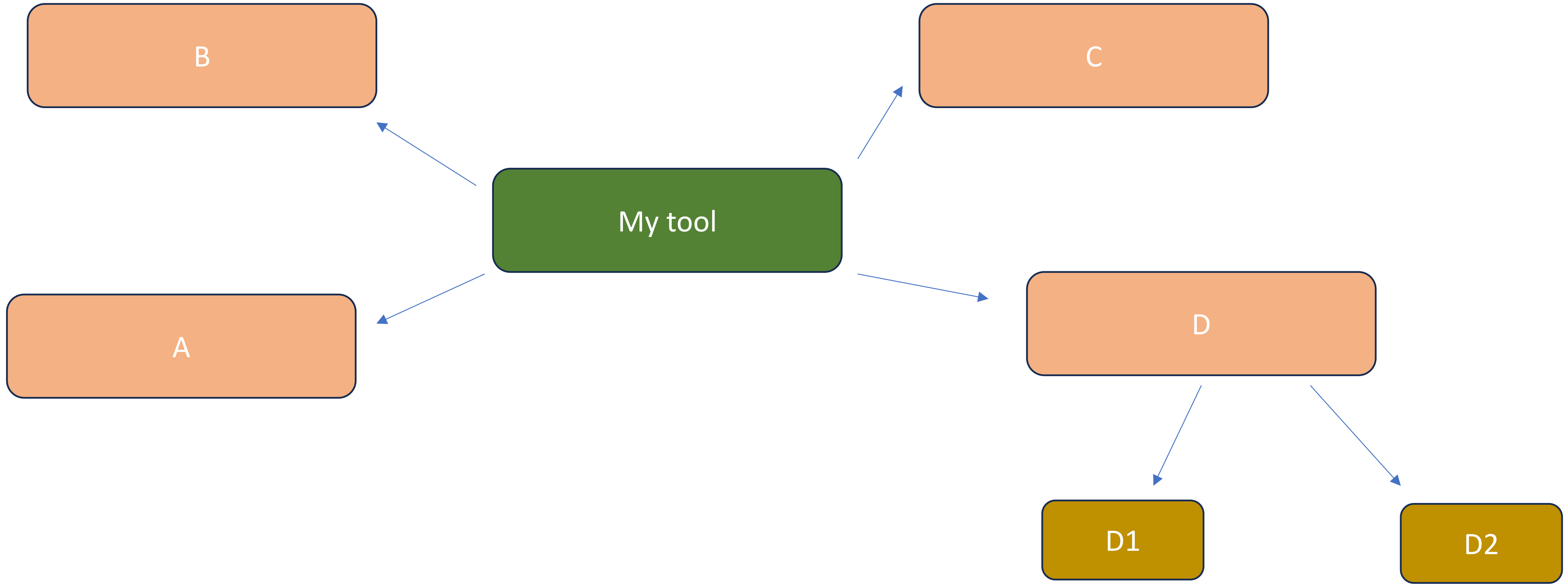
George Githinji

KEMRI-Wellcome Trust Research Programme

NGS course March 2024


Research
Data
Software
Manuscript

# Tools and dependencies



## Exercise1: Installing software

```
#Navigate to https://github.com/lh3/bwa

docker pull alpine
docker images
docker run –it image_id

git clone https://github.com/lh3/bwa.git
cd bwa
make

apk add git
apk add gcc
apk add libc-dev
apk add zlib-dev
make
```

## How can we address dependencies?

- Tarballs
- Package managers
- Virtual images
- Containers

# Tarball

- Code
- Configuration file
- Scripts to compile
- Scripts to install the code

- Time to install
- Conflicts with already installed tools

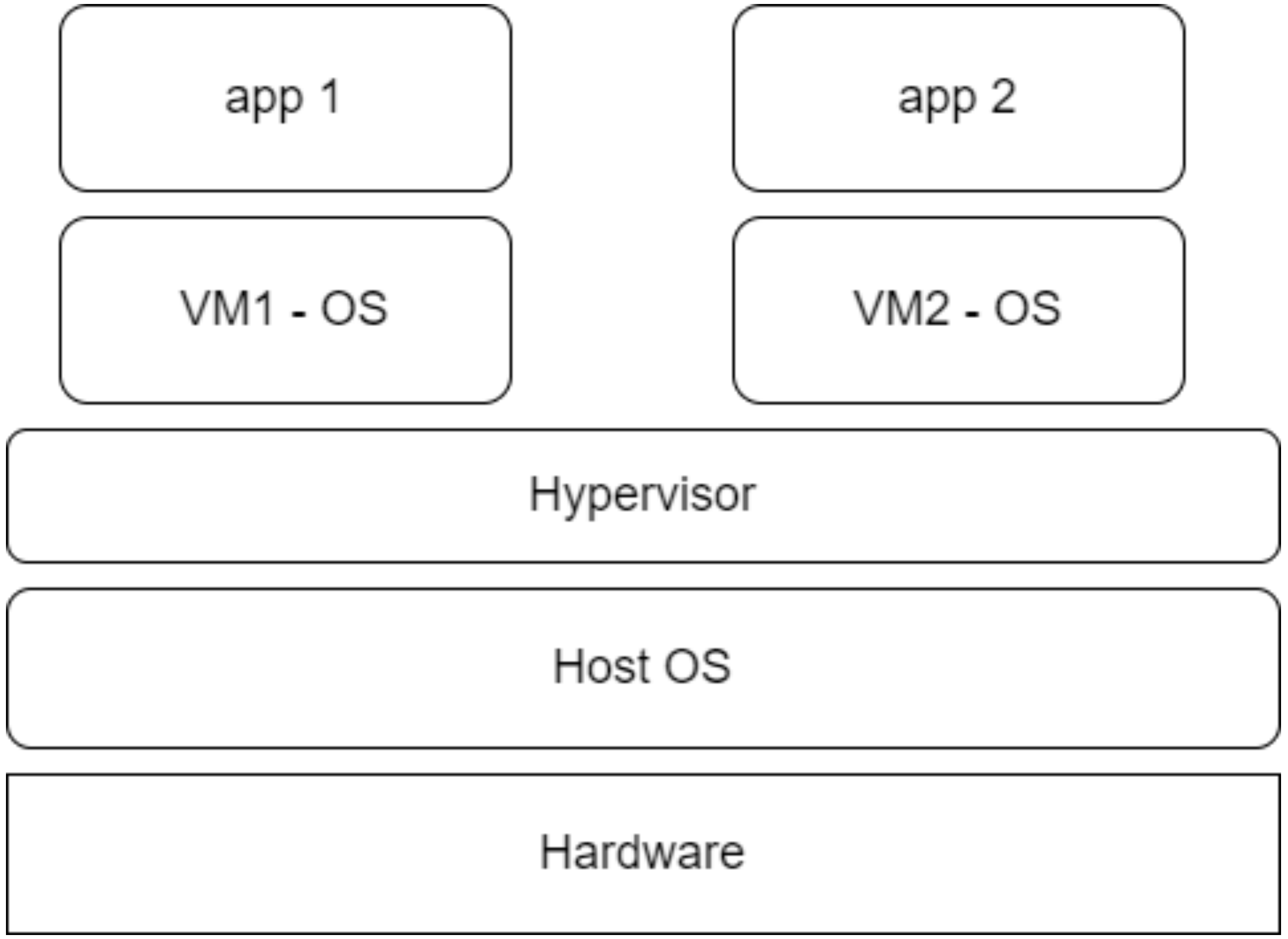
## Package managers

- Apt, yum, apk,

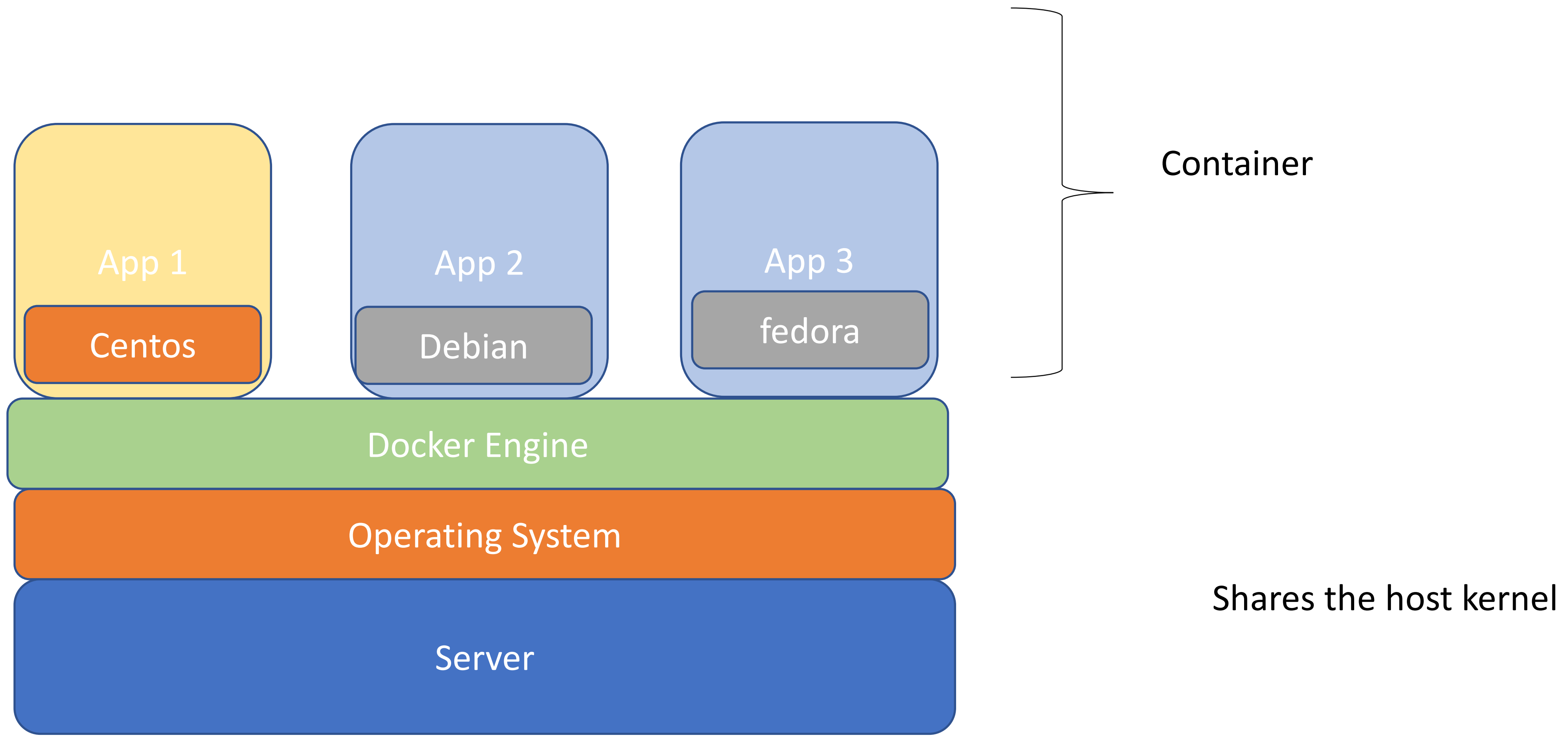
- Perl – cpan
- Python – conda, pip, pip3

- Homebrew

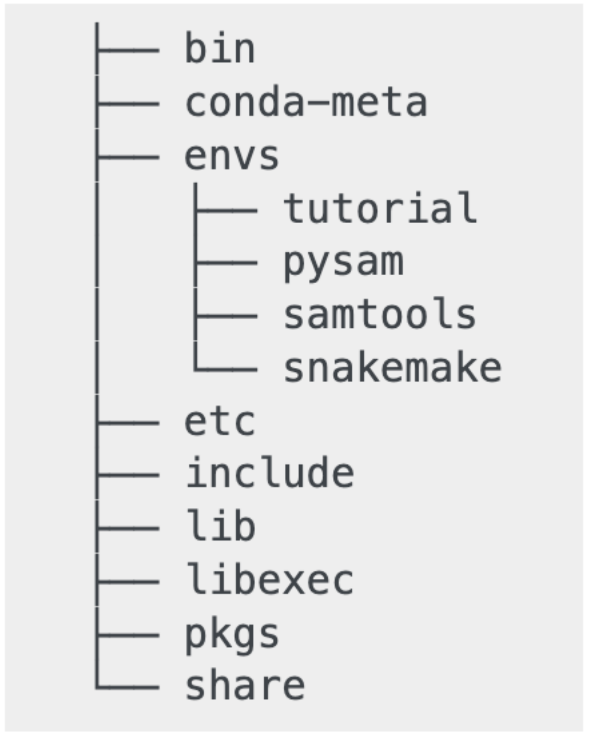
# Virtual machines



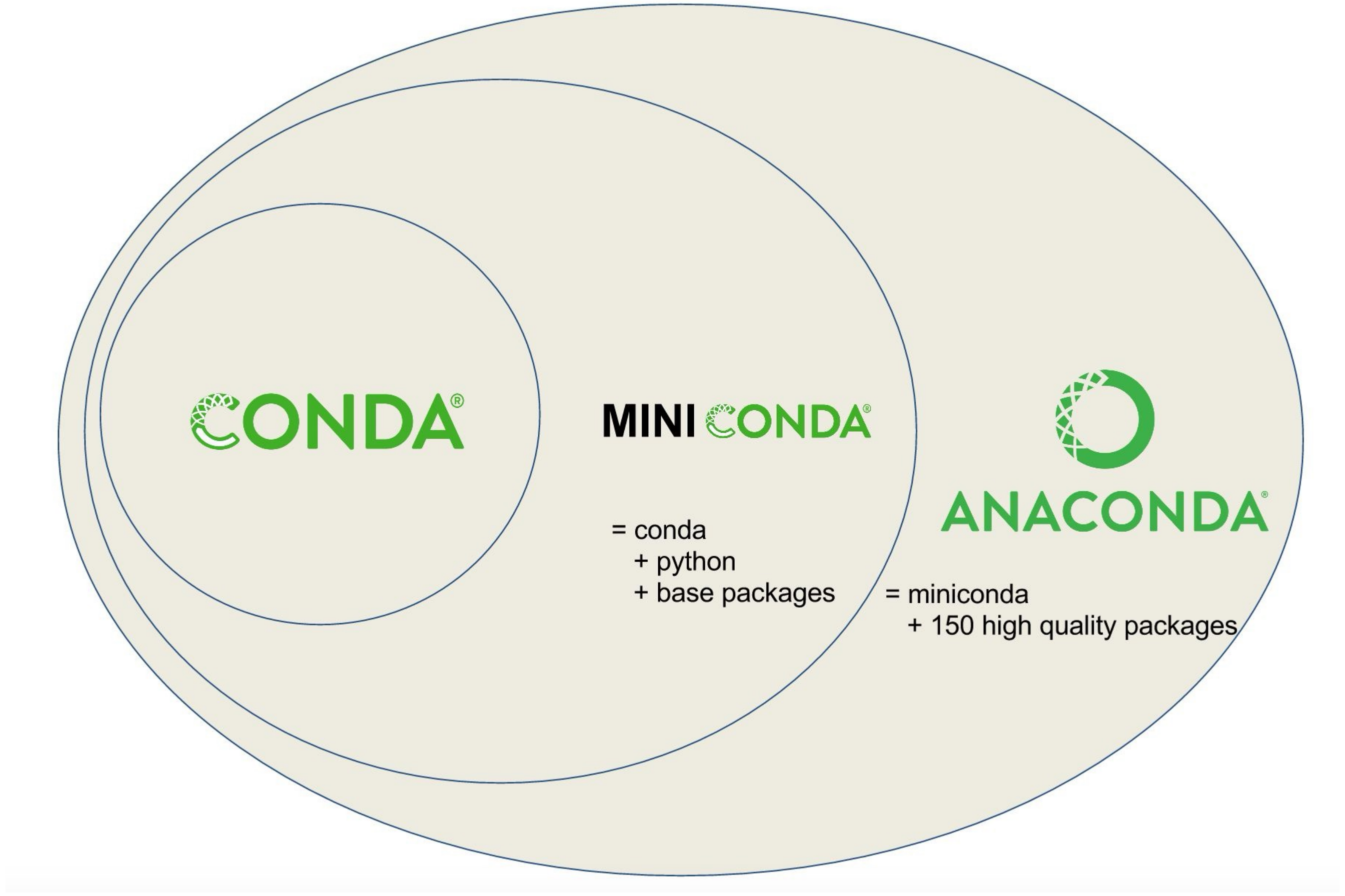
# Containers


Container
App 1
Centos
App 2
Debian
App 3
fedora
Docker Engine
Operating System
Server
Shares the host kernel

## Conda



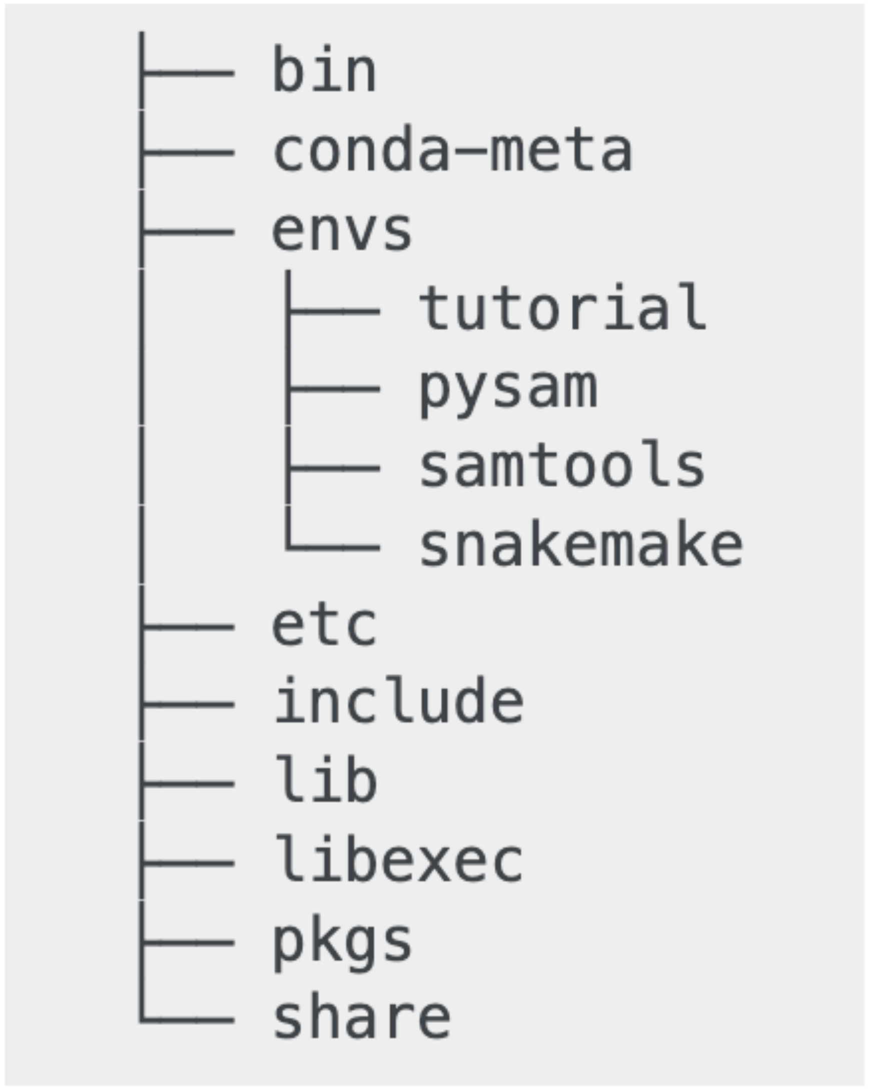
```
├── bin
├── conda-meta
├── envs
│   ├── tutorial
│   ├── pysam
│   ├── samtools
│   └── snakemake
├── etc
├── include
├── lib
├── libexec
├── pkgs
└── share
```



Courtesy: https://bioconda.github.io/tutorials/gcb2020.html#what-exactly-is-conda

## Exercise 2: Installing conda

```
mkdir -p ~/miniconda3

wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh -O
~/miniconda3/miniconda.sh

bash ~/miniconda3/miniconda.sh -b -u -p ~/miniconda3

rm -rf ~/miniconda3/miniconda.sh

~/miniconda3/bin/conda init bash

# Install Mamba via Miniforge

curl -L -O https://github.com/conda-
forge/miniforge/releases/latest/download/Miniforge3-$(uname)-$(uname -m).sh

bash Miniforge3-$(uname)-$(uname –m).sh
```



```
├── bin
├── conda-meta
├── envs
│   ├── tutorial
│   ├── pysam
│   ├── samtools
│   └── snakemake
├── etc
├── include
├── lib
├── libexec
├── pkgs
└── share
```

Courtesy:

## Exercise 3: Creating a conda enviroment

```
conda create -n new-env
conda activate new-env
conda deactivate

conda create -n python-v3.2 python=3.2
conda activate python-v3.2
conda deactivate

conda activate basepython –version
conda activate new-env
python --version  # same as base

conda activate python-v3.2
python --version # different than base#

remove the enviroment
conda remove -n python-v3.2
```

## Installing bwa via conda

```
conda create -n bwa-env
conda activate bwa-env

conda config --show channels

conda config --prepend channels bioconda
conda config --prepend channels conda-forge

conda install bwa

# conda install –c bioconda bwa

conda deactivate
```

```
#Display help for conda command (and sub-commands):
conda --help
conda list --help

#List packages in current conda environment:
$ conda list

#Display all information about conda installation:
conda info -a

#Search for available packages (using regular expressions):
conda search '^doc'   # packages that start with "doc"

# Update package:
conda update wxpython

#Uninstall package:
conda remove wxpython
```

## Sharing Environments: additional resources

https://carpentries-incubator.github.io/introduction-to-conda-for-data-scientists/04-sharing-environments/index.html

```
name: pangolin
channels:
  - conda-forge
  - bioconda
  - defaults
dependencies:
  - biopython>=1.74
  - minimap2>=2.16
  - pip=19.3.1
  - python>=3.7
  - snakemake-minimal=7.24.0
  - gofasta
  - ucsc-fatovcf>=426
  - usher>=0.5.4
  - git-lfs
  - pip:
    - git+https://github.com/cov-lineages/pangolin-data.git
    - git+https://github.com/cov-lineages/scorpio.git
    - git+https://github.com/cov-lineages/constellations.git
```

```
conda env create -f environment.yml
```

## Common Issues and Troubleshooting

- **Dependency Conflicts**:  YAML files might not handle complex dependency scenarios perfectly. In such cases, consider using environment-solving tools like mamba or conda-forge.

- **Environment Activation:** If you encounter issues activating your environment, ensure that your shell supports Conda. You might need to initialize Conda in your shell configuration.

- **Channel Priorities**: Understand the order of channels in your YAML file. Channels listed first take precedence. This is crucial when mixing channels like defaults and conda-forge.

- **PackageNotFound:** Double-check the package names and versions in your YAML file. Ensure the correct channels are specified.

- **UnsatisfiableError:** Adjust version constraints in your YAML file to find a compatible set of packages.

# Pros and cons of Conda

- Pros:
  - Reproducibility: Easily recreate environments on different machines.
  - Shareability: Share YAML files for consistent environments among collaborators.
- Cons:
  - Size: Environments can be large, especially with complex dependencies.
  - Compatibility Issues: Some packages may have dependencies that conflict with others, requiring careful management.