CONTAINERIZATION

Alexander Moreno Briceño

UNIVERSIDAD ANTONIO NARIÑO

HSF
HEP Software Foundation

S³ School — Sustainable Scientific Software School

"Good coding practices to develop better software for your research"

14–21 January 2026 at LAPP, Annecy, France

January 20, 2026

Some Questions...

- Are you familiar with containers in general?

Some Questions...

- Are you familiar with containers in general?
- Are you familiar with containers in software and computing?

Some Questions...

- Are you familiar with containers in general?
- Are you familiar with containers in software and computing?
- Do you know what a Dockerfile is?

Some Questions...

- Are you familiar with containers in general?
- Are you familiar with containers in software and computing?
- Do you know what a Dockerfile is?
- Have you written a Dockerfile before?

Some Questions...

- Are you familiar with containers in general?
- Are you familiar with containers in software and computing?
- Do you know what a Dockerfile is?
- Have you written a Dockerfile before?
- Do you know what a container image is?

Some Questions...

- Are you familiar with containers in general?
- Are you familiar with containers in software and computing?
- Do you know what a Dockerfile is?
- Have you written a Dockerfile before?
- Do you know what a container image is?
- Have you built and run a container?

Some Questions...

- Are you familiar with containers in general?
- Are you familiar with containers in software and computing?
- Do you know what a Dockerfile is?
- Have you written a Dockerfile before?
- Do you know what a container image is?
- Have you built and run a container?
- Have you run code inside a container?

Previously...

Objective of this lesson

- Introduce containers for creating and maintaining reproducible software environments to ensure reproducible research results

Objective of this lesson

- Introduce containers for creating and maintaining reproducible software environments to ensure reproducible research results
  - Docker/Podman

Objective of this lesson

- Introduce containers for creating and maintaining reproducible software environments to ensure reproducible research results
    - Docker/Podman
    - Apptainer (for HPC/HTC systems)

Activity: The PKoffee Container

- Write the PKoffee Dockerfile
- Build the image

```
docker build -t <USERNAME/CONTAINER_NAME> .
```

- Run the Container from the Image

```
docker run -it -p 8000:8000 <USERNAME/CONTAINER_NAME>
```

- Run the Analysis

```
pkoffee analyze - -data-file analysis/coffee_productivity.csv - -output analysis/fitted_models.toml - -show-rankings
```

- Generate the Plot

```
pkoffee plot - -data-file analysis/coffee_productivity.csv - -models analysis/fitted_models.toml - -output analysis/productivity_plot.png
```

- Start a Web Server in the Container

```
python3 -m http.server
```

Outline

1. Introducing Containers

Outline

1. Introducing Containers

2. Docker/Podman Command Line

Outline

1. Introducing Containers

2. Docker/Podman Command Line

3. Removal of Containers and Images

Outline

1. Introducing Containers
2. Docker/Podman Command Line
3. Removal of Containers and Images
4. Finding Containers on Docker Hub

Outline

1. Introducing Containers
2. Docker/Podman Command Line
3. Removal of Containers and Images
4. Finding Containers on Docker Hub
5. Writing Dockerfiles and Building Images

Outline

1. Introducing Containers
2. Docker/Podman Command Line
3. Removal of Containers and Images
4. Finding Containers on Docker Hub
5. Writing Dockerfiles and Building Images
6. Apptainer

Outline

1. Introducing Containers
2. Docker/Podman Command Line
3. Removal of Containers and Images
4. Finding Containers on Docker Hub
5. Writing Dockerfiles and Building Images
6. Apptainer
7. Activity: The PKoffee Container

Introducing Containers

- What are containers?

Introducing Containers

- What are containers?
- Why are they important?

Introducing Containers

- What are containers?
- Why are they important?
- What is Docker, Podman and Apptainer?

Introducing Containers
Scientific Software and Reproducibility Challenges

- Difficulty installing software tools due to multiple dependencies.

Introducing Containers
Scientific Software and Reproducibility Challenges

- Difficulty installing software tools due to multiple dependencies.
- Software unavailable for some OS.

Introducing Containers
Scientific Software and Reproducibility Challenges

- Difficulty installing software tools due to multiple dependencies.
- Software unavailable for some OS.
- Inconsistent results in collaborations due to different software versions or OS.

Scientific Software and Reproducibility Challenges

- Difficulty installing software tools due to multiple dependencies.
- Software unavailable for some OS.
- Inconsistent results in collaborations due to different software versions or OS.
- Software works only on your computer.

Introducing Containers
Scientific Software and Reproducibility Challenges

- Difficulty installing software tools due to multiple dependencies.
- Software unavailable for some OS.
- Inconsistent results in collaborations due to different software versions or OS.
- Software works only on your computer.
- Hard to install your software package on other computers or OS.

Introducing Containers
Scientific Software and Reproducibility Effects

- Cannot replicate software environments.

Introducing Containers
Scientific Software and Reproducibility Effects

- Cannot replicate software environments.
- Cannot use tools because they are unavailable or hard to install.

Introducing Containers
Scientific Software and Reproducibility Effects

- Cannot replicate software environments.
- Cannot use tools because they are unavailable or hard to install.
- Cannot build upon work because system configuration cannot be reproduced.

Introducing Containers
Scientific Software and Reproducibility Effects

- Cannot replicate software environments.
- Cannot use tools because they are unavailable or hard to install.
- Cannot build upon work because system configuration cannot be reproduced.
- Cannot reproduce results.

Introducing Containers
Scientific Software and Reproducibility Effects

- Cannot replicate software environments.
- Cannot use tools because they are unavailable or hard to install.
- Cannot build upon work because system configuration cannot be reproduced.
- Cannot reproduce results.

Containers to the Rescue!

Package software, dependencies, and resources into a consistent, reproducible and portable environment.

Introducing Containers

A very large metal box used for transporting goods

https://dictionary.cambridge.org/dictionary/english/container

Introducing Containers

A very large metal box used for transporting goods by rail, sea, air and/or road

Google Slides

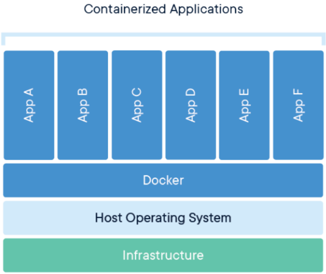
Introducing Containers

A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another

https://www.docker.com/resources/what-container/

Introducing Containers

A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another



https://www.docker.com/resources/what-container/

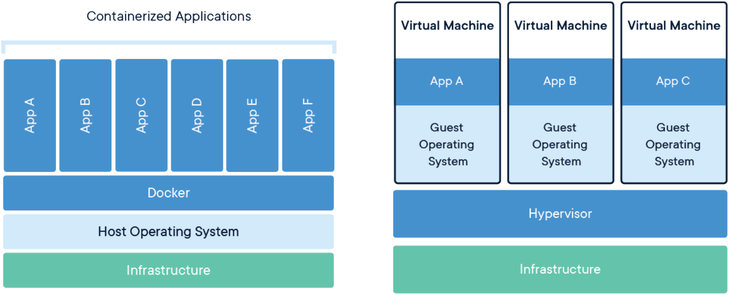
Introducing Containers

Containers and Virtual Machines

https://www.docker.com/resources/what-container/

Introducing Containers

Containers and Virtual Machines



https://www.docker.com/resources/what-container/

Introducing Containers

https://hsf-training.github.io/hsf-training-docker/01-introduction/index.html

Introducing Containers

- It is a platform for developing, shipping, and running applications in containers.

https://hsf-training.github.io/hsf-training-docker/01-introduction/index.html

Introducing Containers

- It is a platform for developing, shipping, and running applications in containers.
- Docker provides a public registry for sharing and collaborating on container images called Docker Hub

https://hsf-training.github.io/hsf-training-docker/01-introduction/index.html

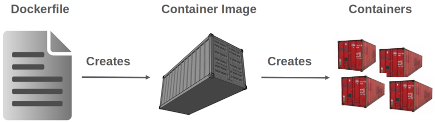
Introducing Containers

- It is a platform for developing, shipping, and running applications in containers.
- Docker provides a public registry for sharing and collaborating on container images called Docker Hub
- Docker has very similar syntax to Git and Linux

https://hsf-training.github.io/hsf-training-docker/01-introduction/index.html

Introducing Containers

- It is a platform for developing, shipping, and running applications in containers.
- Docker provides a public registry for sharing and collaborating on container images called Docker Hub
- Docker has very similar syntax to Git and Linux
- Docker images are executables that bundle together all necessary components for an application or an environment. Docker containers are the runtime instances of images



https://hsf-training.github.io/hsf-training-docker/01-introduction/index.html

Introducing Containers

https://hsf-training.github.io/hsf-training-docker/01-introduction/index.html

Introducing Containers

- Podman is an open-source alternative to Docker with several advantages:

`https://hsf-training.github.io/hsf-training-docker/01-introduction/index.html`

Introducing Containers

- Podman is an open-source alternative to Docker with several advantages:
  - Podman is able to run containers as a non-root user out of the box, a big security advantage over Docker. Podman uses a servless architecture, and it does not require a daemon running as superuser to execute containers as Docker does

https://hsf-training.github.io/hsf-training-docker/01-introduction/index.html

Introducing Containers

podman

- Podman is an open-source alternative to Docker with several advantages:
  - Podman is able to run containers as a non-root user out of the box, a big security advantage over Docker. Podman uses a servless architecture, and it does not require a daemon running as superuser to execute containers as Docker does
  - Docker Desktop has licensing restrictions that may prevent you from using it in some institutions

https://hsf-training.github.io/hsf-training-docker/01-introduction/index.html

Introducing Containers

- Podman is an open-source alternative to Docker with several advantages:
  - Podman is able to run containers as a non-root user out of the box, a big security advantage over Docker. Podman uses a servless architecture, and it does not require a daemon running as superuser to execute containers as Docker does
  - Docker Desktop has licensing restrictions that may prevent you from using it in some institutions
  - Podman is a drop-in replacement for Docker, so you can use the same commands and workflows you are used to with Docker

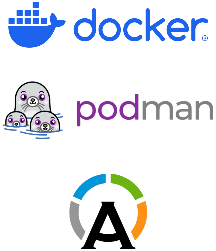
https://hsf-training.github.io/hsf-training-docker/01-introduction/index.html

Introducing Containers

Introducing Containers

- Apptainer (formerly known as Singularity) is another containerization technology
- Used widely in HPC

Introducing Containers
Containers Advantages for Scientific Software



- Reproducibility
- Portability
- Scalability
- Configurability
- Documentation
- Preservation
- Distribution

Docker/Podman Command Line

docker®

podman

Docker/Podman Command Line

docker

podman

- Version and Info

```
docker - -version
docker version
docker info
```

```
podman - -version
podman version
docker info
```

Docker/Podman Command Line

docker | podman

- Version and Info

```
docker - -version
docker version
docker info
```

```
podman - -version
podman version
docker info
```

- List of Images

```
docker images
docker image ls
docker images <repository name>
docker images - -filter=reference="word"
```

```
podman images
podman image ls
podman images <repository name>
podman images - -filter=reference="word"
```

Docker/Podman Command Line

- Pulling an Image

```
docker pull <image name>
```

```
podman pull <image name>
```

Docker/Podman Command Line

- Pulling an Image

```
docker pull <image name>
```

```
podman pull <image name>
```

Exercise

Pull the image python:3.9-slim for Python 3.9 and then list all Python images

- Running an Image

```
docker run <image name>
docker run -it <image name> /bin/bash
docker container run -it <image name> /bin/bash
```

```
podman run <image name>
podman run -it <image name> /bin/bash
podman container run -it <image name> /bin/bash
```

- Running an Image

```
docker run <image name>
docker run -it <image name> /bin/bash
docker container run -it <image name> /bin/bash
```

```
podman run <image name>
podman run -it <image name> /bin/bash
podman container run -it <image name>
/bin/bash
```

- List of Containers

```
docker ps
docker container ps
docker container list
docker container ls
docker container ls - -all
```

```
podman ps
podman container ps
podman container list
podman container ls
podman container ls - -all
```

Removal of Containers and Images

- Removing an Image

```
docker image rm <image name>
docker image rm <image id>
```

```
podman image rm <image name>
podman image rm <image id>
```

Removal of Containers and Images

- Removing an Image

```
docker image rm <image name>
docker image rm <image id>
```

```
podman image rm <image name>
podman image rm <image id>
```

Exercise

Start an instance of a container, exit it, and then remove it with docker/podman rm

Removal of Containers and Images

- Removing an Image

```
docker image rm <image name>
docker image rm <image id>
```

```
podman image rm <image name>
podman image rm <image id>
```

Exercise

Start an instance of a container, exit it, and then remove it with docker/podman rm

- Removing Containers

```
docker rm <container name>
docker rm <container id>
```

```
podman rm <container name>
podman rm <container id>
```

Finding Containers on Docker Hub

Docker Hub

Naming convention for Docker images

```
OWNER/CONTAINER_IMAGE_NAME:TAG
```

Writing Dockerfiles

Why create your own image?

- Customization: No existing container image with all the tools you need
- Reproducibility: Archive specific versions of a project
- Collaboration: Share a unified workflow

Writing Dockerfiles

Why create your own image?

- Customization: No existing container image with all the tools you need
- Reproducibility: Archive specific versions of a project
- Collaboration: Share a unified workflow

Interactive Installation

```
podman container run -it alpine
```

Writing Dockerfiles

A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image.

https://docs.docker.com/reference/dockerfile/

https://docs.docker.com/build/building/best-practices/

Writing Dockerfiles

A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image.

Dockerfile

```
FROM <Existing Image>
RUN <Install CMDs from Shell>
CMD <CMD to Run by Default>
```

https://docs.docker.com/reference/dockerfile/

https://docs.docker.com/build/building/best-practices/

Building and Running Images

- Building Images

```
docker image build -f Dockerfile -t
USERNAME/CONTAINER_IMAGE_NAME .

docker image build -t USERNAME/CONTAINER_IMAGE_NAME .
```

```
podman image build -f Dockerfile -t
USERNAME/CONTAINER_IMAGE_NAME .

podman image build -t USERNAME/CONTAINER_IMAGE_NAME .
```

Building and Running Images

- Building Images

```
docker image build -f Dockerfile -t USERNAME/CONTAINER_IMAGE_NAME .

docker image build -t USERNAME/CONTAINER_IMAGE_NAME .
```

```
podman image build -f Dockerfile -t USERNAME/CONTAINER_IMAGE_NAME .

podman image build -t USERNAME/CONTAINER_IMAGE_NAME .
```

- Pushing Images

```
docker image push USERNAME/CONTAINER_IMAGE_NAME
```

```
podman image push USERNAME/CONTAINER_IMAGE_NAME
```

Apptainer

- Apptainer (formerly known as Singularity) is a free and open-source container platform that allows you to create and run applications in isolated environments in a simple, portable, fast, and secure manner
- It performs operating system level virtualization known as containerization
- It is designed to bring containers and reproducibility to the scientific community and High-Performance Computing (HPC) use cases

https://mambelli.github.io/hsf-training-singularity-webpage/

Apptainer

- It allows to build and run containers with just a few steps in most of the cases, and its design presents key advantages for the scientific community:
  - Single-file based container images, facilitating the distribution, archiving and sharing
  - Ability to run, and in modern systems also to be installed, without any privileges
  - Preserves the permissions in the environment. The user outside the container can be the same user inside
  - Simple integration with resource managers and distributed computing frameworks because it runs as a regular application
  - Minimum overhead. No extra processes after initializing the container

`https://mambelli.github.io/hsf-training-singularity-webpage/`

Images and Containers

- Pulling and Running Images

```
apptainer pull hello-world.sif shub://vsoch/hello-world
apptainer run hello-world.sif
```

- Image Cache

```
apptainer cache list -v
```

- Running Commands in a Container

```
apptainer exec hello-world.sif /bin/echo Bon Jour!
apptainer exec hello-world.sif /bin/sh
apptainer shell hello-world.sif
```

Images and Containers

- Docker Images

```
apptainer pull python-3.9.6.sif docker://python:3.9.6-slim-buster
```

- Building Images

```
apptainer build filename.sif filename.def
```

https://apptainer.org/docs/user/main/definition_files.html

Activity: The PKoffee Container

- Write the PKoffee Dockerfile
- Build the image

```
docker build -t <USERNAME/CONTAINER_NAME> .
```

- Run the Container from the Image

```
docker run -it -p 8000:8000 <USERNAME/CONTAINER_NAME>
```

- Run the Analysis

```
pkoffee analyze - -data-file analysis/coffee_productivity.csv - -output analysis/fitted_models.toml - -show-rankings
```

- Generate the Plot

```
pkoffee plot - -data-file analysis/coffee_productivity.csv - -models analysis/fitted_models.toml - -output analysis/productivity_plot.png
```

- Start a Web Server in the Container

```
python3 -m http.server
```