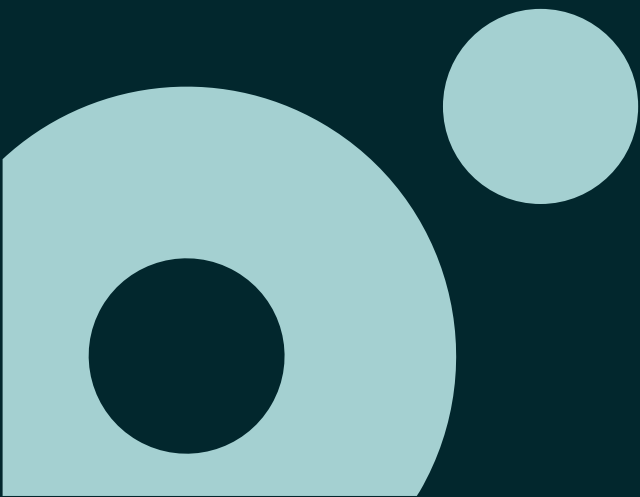# Containers & Kubernetes
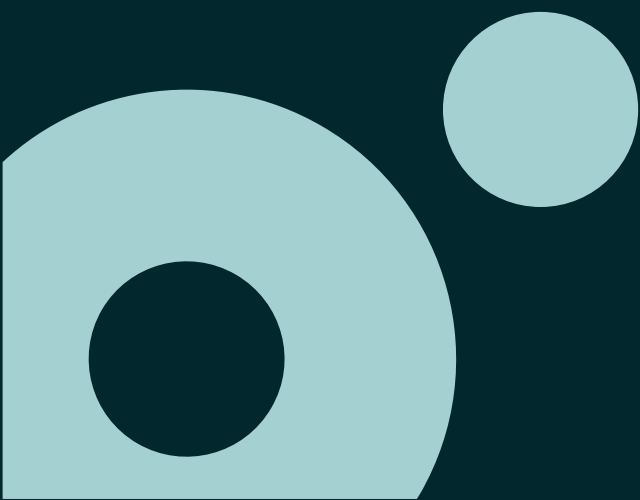## Session #03

moOngy.

Storage

Environment Variables

CMD vs ENTRYPOINT

Limits

Lab

Storage

Persistent Storage for containers
Storage

- Data doesn't persist when a container is removed, and it can be difficult to get the data out of the container if another process needs it

- A container's writable layer is tightly coupled to the host machine where the container is running

- Writing into a container's writable layer requires a storage driver to manage the filesystem

moOngy.

Mounting data in containers
Storage

- tmpfs mounts
  - Stores data in host system memory ONLY (Linux-only)
  - Data is not created on container writable layer

- bind mounts
  - File or directory on the host machine is mounted into a container
  - Referenced by its absolute path on the host machine

- Volumes
  - Preferred mechanism for persisting data generated by and used by Docker containers
  - Stored in a part of the host filesystem which is managed by Docker

moOngy.

When to use tempfs
Storage

- Cases when you do not want the data to persist either on the host machine or within the container

- Security reasons or to protect the performance of the container when your application needs to write a large volume

- Don't allow sharing between containers
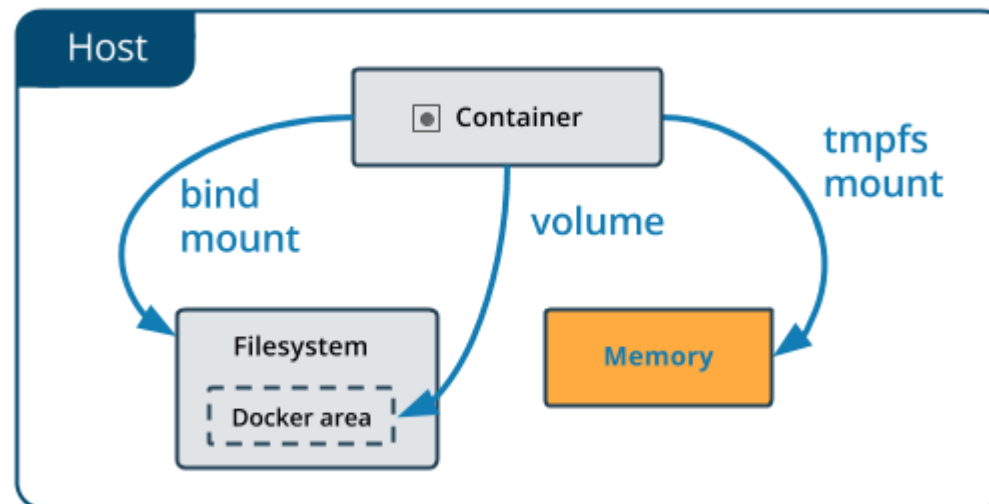
How to use tempfs
Storage

```
docker run -d -it \
--mount type=tmpfs,destination=/tmp  \
nginx:latest
```

- Using **--mount** flag with several options split by comma (**,**)
  - **type=tmpfs**, mandatory and static
  - **destination=folder**, where tmpfs is mounted inside contains
  - **tmpfs-size=# of bytes**, size of tmpfs in bytes. Unlimited by default

moOngy.

How to use tempfs

```
docker run -d -it \
--tmpfs /tmp  \
nginx:latest
```

- Using **--tmpfs** flag
  - Only allows to define destination on the container
  - Exists for legacy purpose
  - **--mount** is the preferable way to mount tmpfs

moOngy.

Bind mounts

Storage

- Very performant, but they rely on the host machine's filesystem
- Hard to share configuration between hosts
- Data already on container is not propagated to the host
- Non-empty directory on the container, the directory's existing contents are obscured by the bind mount
- Allow sharing between containers

moOngy.
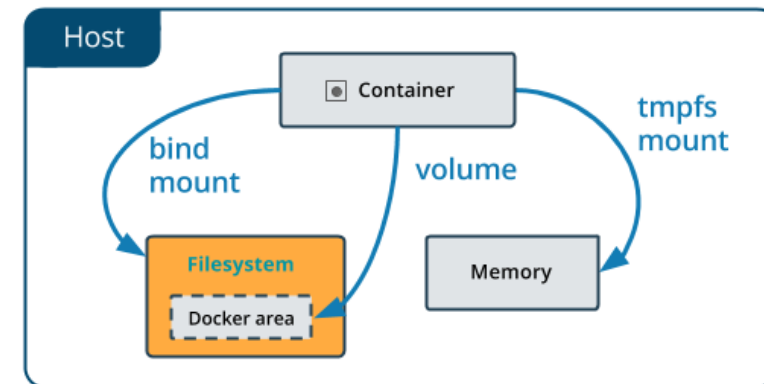
When to use bind mounts

Storage

- Sharing configuration files from the host machine to containers (ex. DNS resolution)

- Sharing source code or build artifacts between a development environment on the Docker host and a container (ex. Debugger)

- When the file or directory structure of the Docker host is guaranteed to be consistent with the bind mounts the containers require.

moOngy.

How to use bind mounts

```
docker run -d -it \
--mount type=bind,source=/tmp/data,target=/share\
nginx:latest
```

- Using **--mount** flag with several options split by comma (**,**)
  - **type=bind**, mandatory and static
  - **source=folder**, host folder to share
  - **target=folder**, container folder to contain data
  - **readonly**, making the mount read-only

moOngy.

How to use bind mounts

```
docker run -d -it \
-v /tmp/data:/share  \
nginx:latest
```

- Using **-v** flag
    - Source and destination folder split by colon (**:**)
    - Access mode as additional block: **:rw** or **:ro**
    - Default access mode is **:rw**

moOngy.

Volumes

Storage

- Volumes are the preferred way to persist data
- Docker object that make part of daemon configuration
- Data is controlled by Docker even can be stored on a common folder on host filesystem
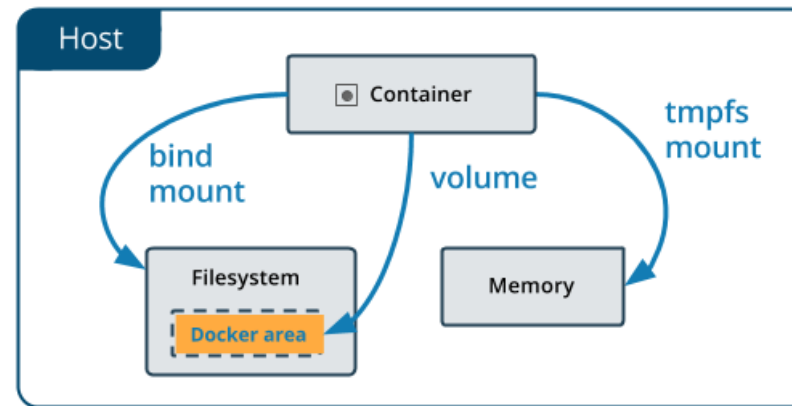
Docker volumes: Advantages
Storage

- Volumes are easier to back up or migrate than bind mounts.

- You can manage volumes using Docker CLI commands or the Docker API.

- Volumes work on both Linux and Windows containers.

- Volumes can be more safely shared among multiple containers.

- Volume drivers let you store volumes on remote hosts or cloud providers, to encrypt the contents of volumes, or to add other functionality.

- New volumes can have their content pre-populated by a container.

moOngy.

When to use volumes
Storage

- Sharing data among multiple running containers

- When the Docker host is not guaranteed to have a given directory or file structure

- When you want to store your container's data on a remote host or a cloud provider, rather than locally

- When you need to back up, restore, or migrate data from one Docker host to another, volumes are a better choice

moOngy.

How to manage Volumes
Storage

- Create a new volume

        docker volume create my-vol

- List all available volumes

        docker volume ls

- Get volume details

        docker volume inspect my-vol

- Delete a volume

        docker volume rm my-vol

moOngy.

How to use volumes

```
docker run -d -it \
--mount source=my-vol,target=/share\
nginx:latest
```

- Using **--mount** flag with several options split by comma (**,**)
  - **source=volume**, volume name to use
  - **target=folder**, container folder to contain data
  - **readonly**, making the mount read-only

moOngy.
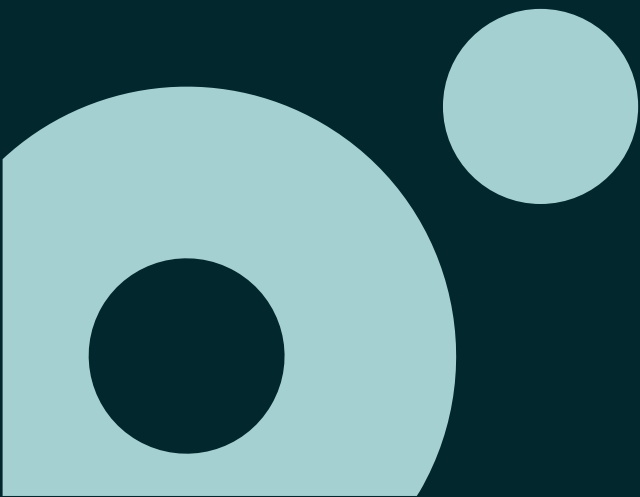
How to use volumes

```
docker run -d -it \
-v my-vol:/share  \
nginx:latest
```

- Using **-v** flag
  - Volume name and destination folder split by colon (**:**)
  - Access mode as additional block: **:rw** or **:ro**
  - Default access mode is **:rw**

moOngy.

moongy. | Demo: Storage

Environment Variables

How to run containers dynamically?
Env Vars

- Container runs on an isolated context

- Environment variables exists on container's scope

- Preferrable way to "send" values to the container allowing different executions

- These variables can be set on docker run command

moOngy.

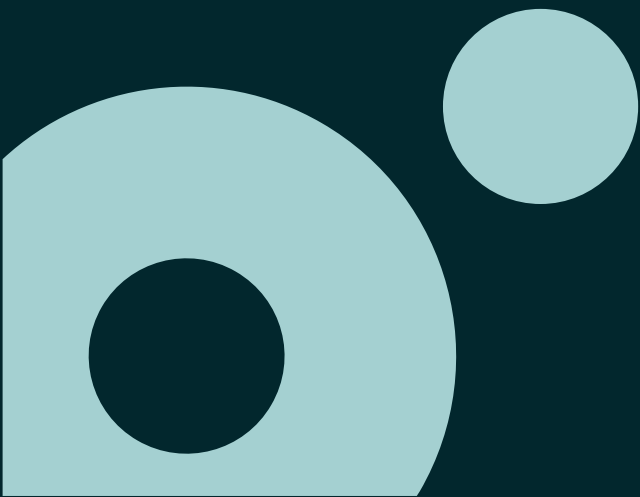# Can be used and set on build time?
Env Vars

- ENV is an available command on Dockerfile

- Using ENV allows to set static values on the image

- ARG is an available command on Dockerfile that allows to dynamically send values during build time

- ARG and ENV can be used together to environment variables on the image

moOngy.

moOngy. | Demo: Env Vars

CMD vs ENTRYPOINT

Shell vs. Exec Form
CMD vs. ENTRYPOINT

- Shell form
  - Runs the commands as a bash command
  - EX. CMD echo "Hello World" ➔ /bin/sh –c 'echo "Hello World"

- Exec form
  - Runs the commands directly without bash context
  - Env vars cannot be used on this approach since only have values inside the shell
  - EX. CMD ["/bin/echo", "Hello", "World"]

- Exec form is preferrable since the process will be executed directly and will be easily controlled regarding container lifecycle

moOngy.

What to choose?
CMD vs. ENTRYPOINT

- CMD
  - Runs a process using arguments set on Dockerfile
  - This command can be overwritten during docker run command

- ENTRYPOINT
  - Runs a process using arguments set on Dockerfile
  - This command cannot be overwritten during docker run command
  - The arguments sent on docker run are sent the process defined on ENTRYPOINT as arguments

- CMD and ENTRYPOINT can work together using CMD values as default arguments for ENTRYPOINT process

moOngy.

moongy. | Demo: Arguments at Run & Build

Limits
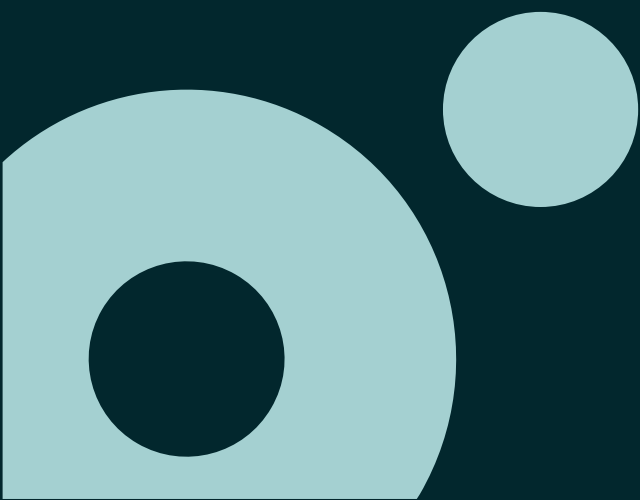
How to restrict resources?
Limits

- Containers shares host resources

- By default, any container can consume all host resources

- Those resources are shared by all containers

- Not having explicit control may cause resources exhaustion causing impact on containers and the host

- When running a container, you may enforce limits on how much CPU and memory can be used by the container

- Reaching the limit will not break the container, only don't allow it to get more resources
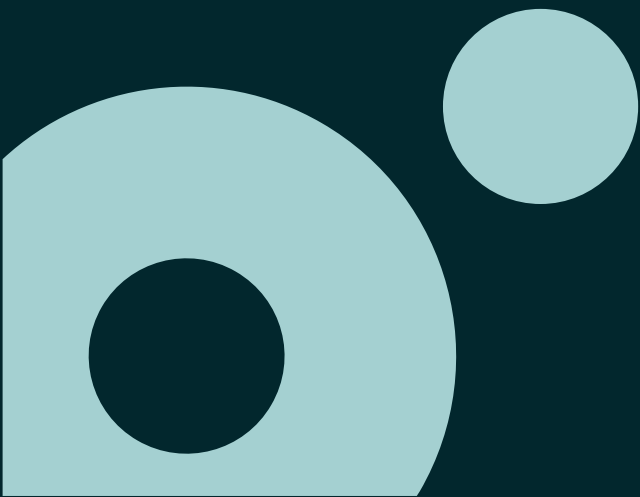
moOngy.

How to restrict resources?
Limits

Option	Description
`--cpus=<value>`	Specify how much of the available CPU resources a container can use. For instance, if the host machine has two CPUs and you set `--cpus="1.5"`, the container is guaranteed at most one and a half of the CPUs. This is the equivalent of setting `--cpu-period="100000"` and `--cpu-quota="150000"`.

Option	Description
`-m` or `--memory=`	The maximum amount of memory the container can use. If you set this option, the minimum allowed value is `6m` (6 megabytes). That is, you must set the value to at least 6 megabytes.

moOngy.

moOngy. | Demo: Limits

Lab

Lab 3: Persistency in containers

Github

[Lab 03 - Persistency in containers | docker-kubernetes-training (tasb.github.io)](#)

moOngy.
Minds on the move

Rua Sousa Martins, nº 10

1050-218 Lisboa | Portugal