Introduction to Docker

What it is and how to use it

LosFuzzys

November 1, 2023

What will be smaller: No one

We prepared some challenges for your.
They can all be solved with commands presented here

- Docker Runner 1
- Docker Runner 2
- Docker Inspector
- Docker Exposer
- Docker Copyer

The name gives a hint on what to do
Also, *Docker Exposer* used information obtained during *Docker Inspector*

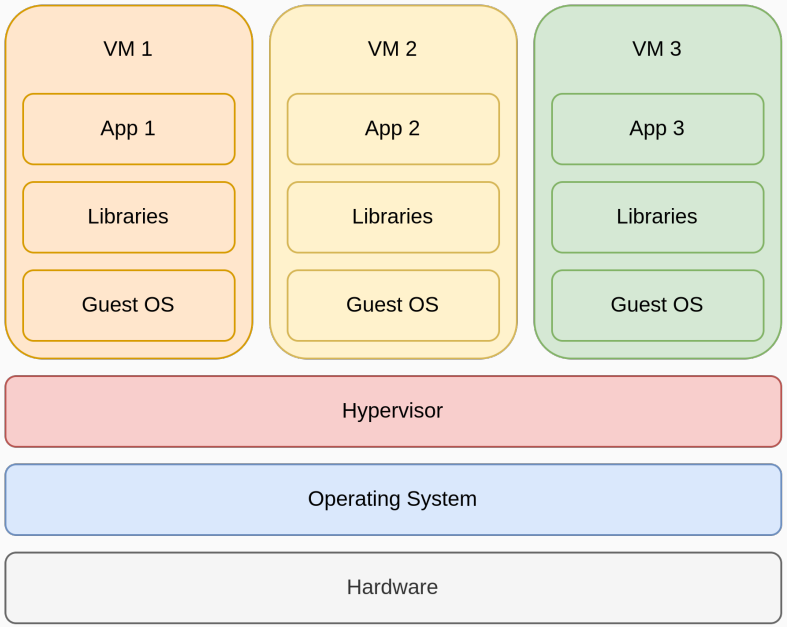
What is Docker?

About Docker

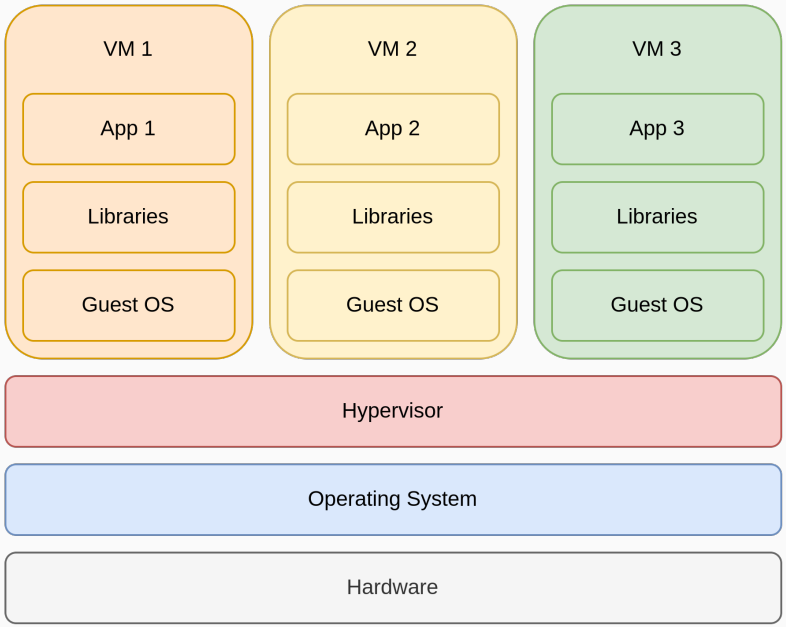
Docker lets us distribute software in containers.
These containers ship with the software and all their dependencies

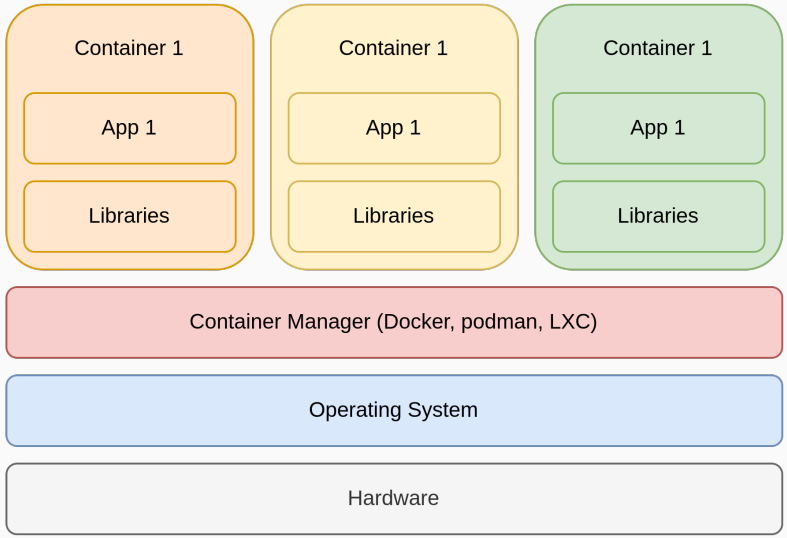
- Easy deployment
- Reproducability (if used right)
- Dependencies between containers can be described easily

Docker vs. VM


VM 1
App 1
Libraries
Guest OS
VM 2
App 2
Libraries
Guest OS
VM 3
App 3
Libraries
Guest OS
Hypervisor
Operating System
Hardware

Docker vs. VM


VM 1
App 1
Libraries
Guest OS
VM 2
App 2
Libraries
Guest OS
VM 3
App 3
Libraries
Guest OS
Hypervisor
Operating System
Hardware


Container 1
App 1
Libraries
Container 1
App 1
Libraries
Container 1
App 1
Libraries
Container Manager (Docker, podman, LXC)
Operating System
Hardware

Benefits and Drawbacks

- No guest OS is needed
- Container are ready as soon as they are spawned
- lightweight if build correctly

- Exploits in the Kernel can be exploited
- Access to the filesystem might be problematic
- Container needs to be build for the hosts OS
- docker.socket

What can run where

Linux container can only run on Linux machines.
But why does Docker-Desktop (for Windows or MacOS) work?

> Under the hood they launch a Linux-vm and this VM runs the container [Fer19]

Where do we use docker?

LosLuzzys uses docker (or containers) everywhere
All our CTFs run on containers

- fuzzy.land
- GlacierCTF
- KaindorfCTF

Also our internal services run in containers

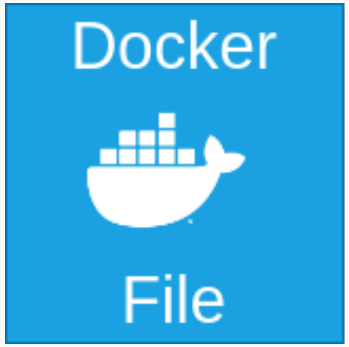
Dockerfile, Image and Container

There are three distinct components:

Dockerfile: describes how the Dockerimage looks

Dockerimage: blueprint for Container

Dockercontainer: The thing that is running



Dockerimages

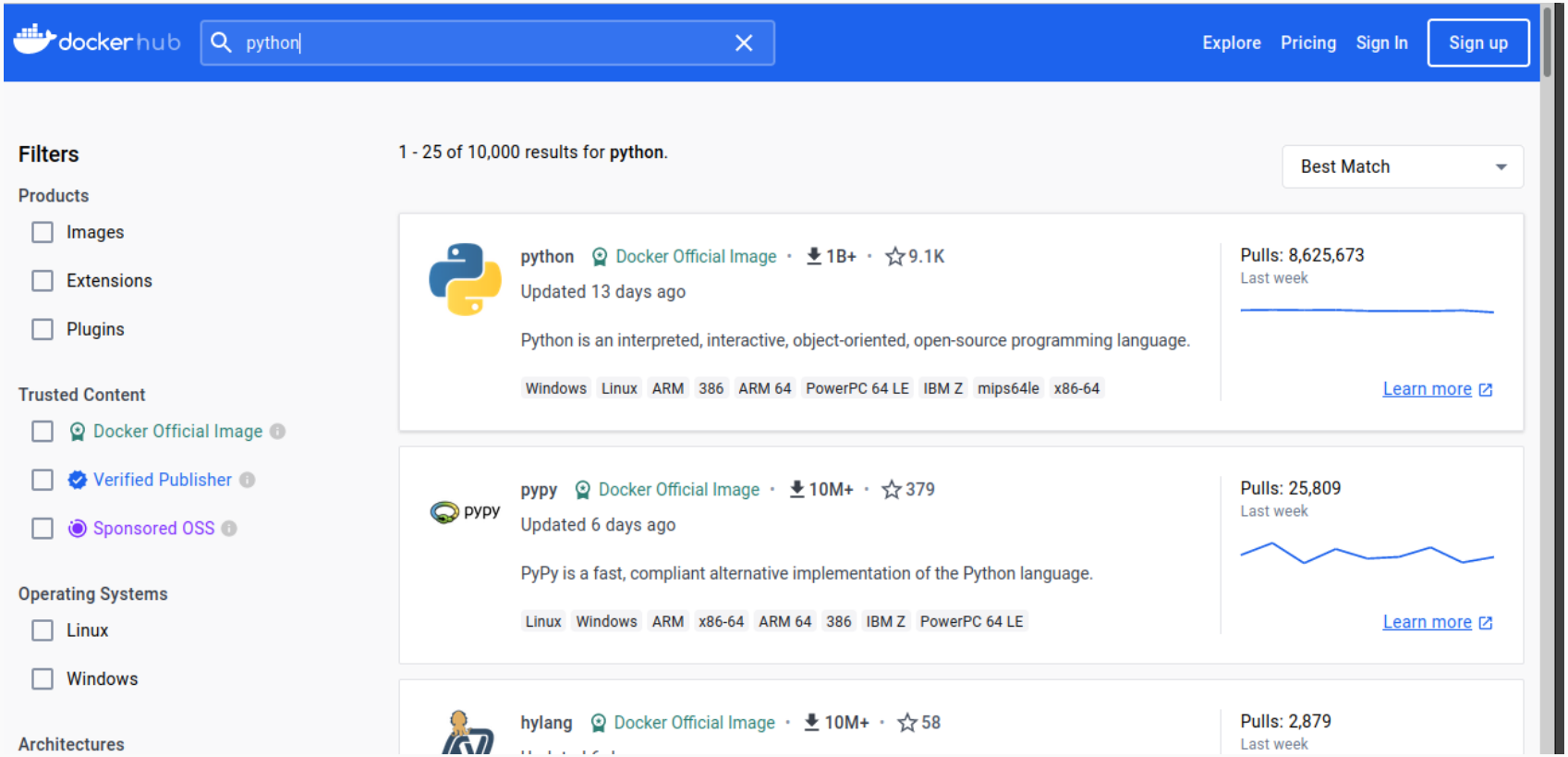
What are images

The Images are the blueprints for the container

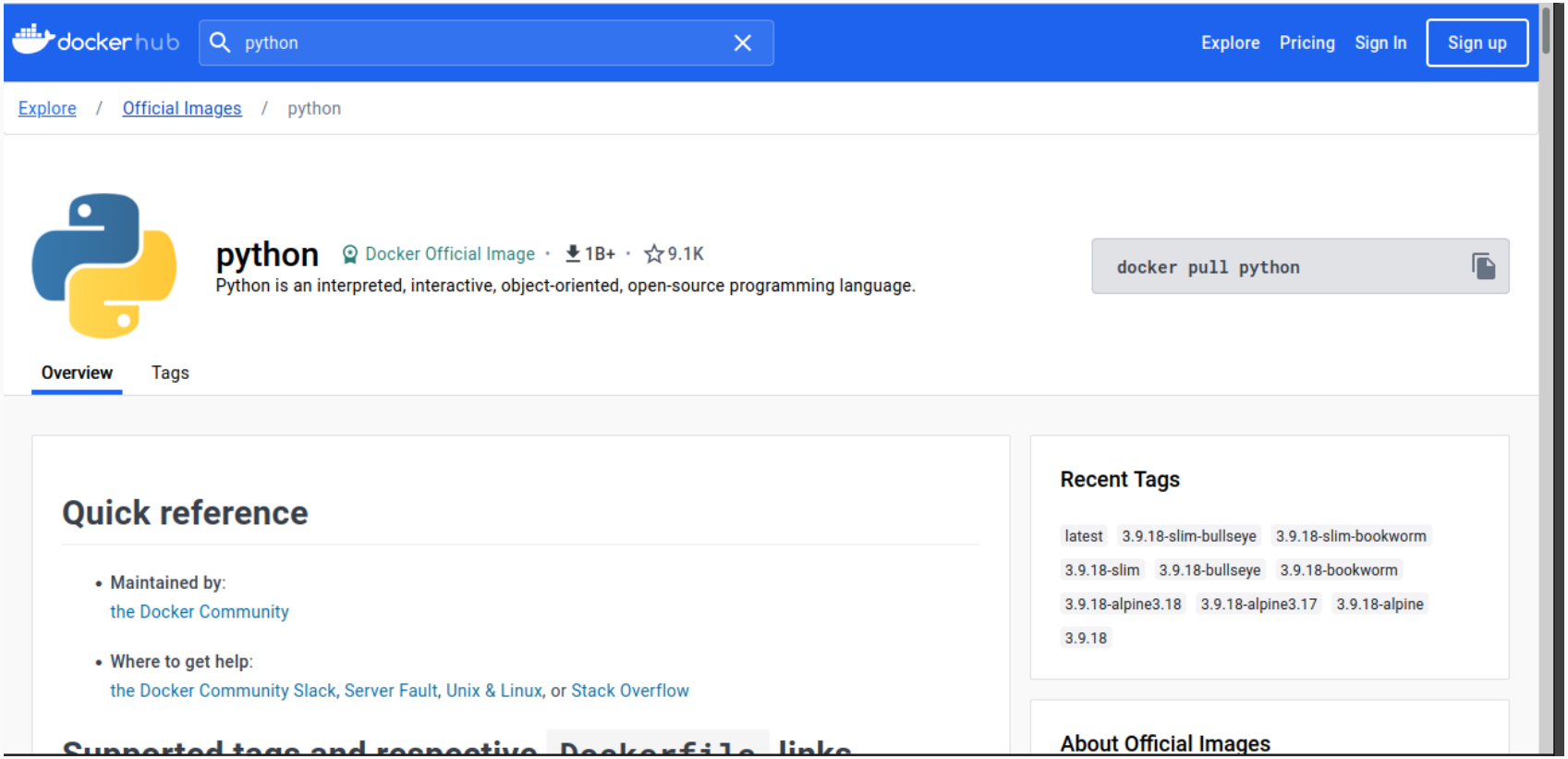
They can be shared with others.

- Dockerhub `https://hub.docker.com/`
- Github Container Registry

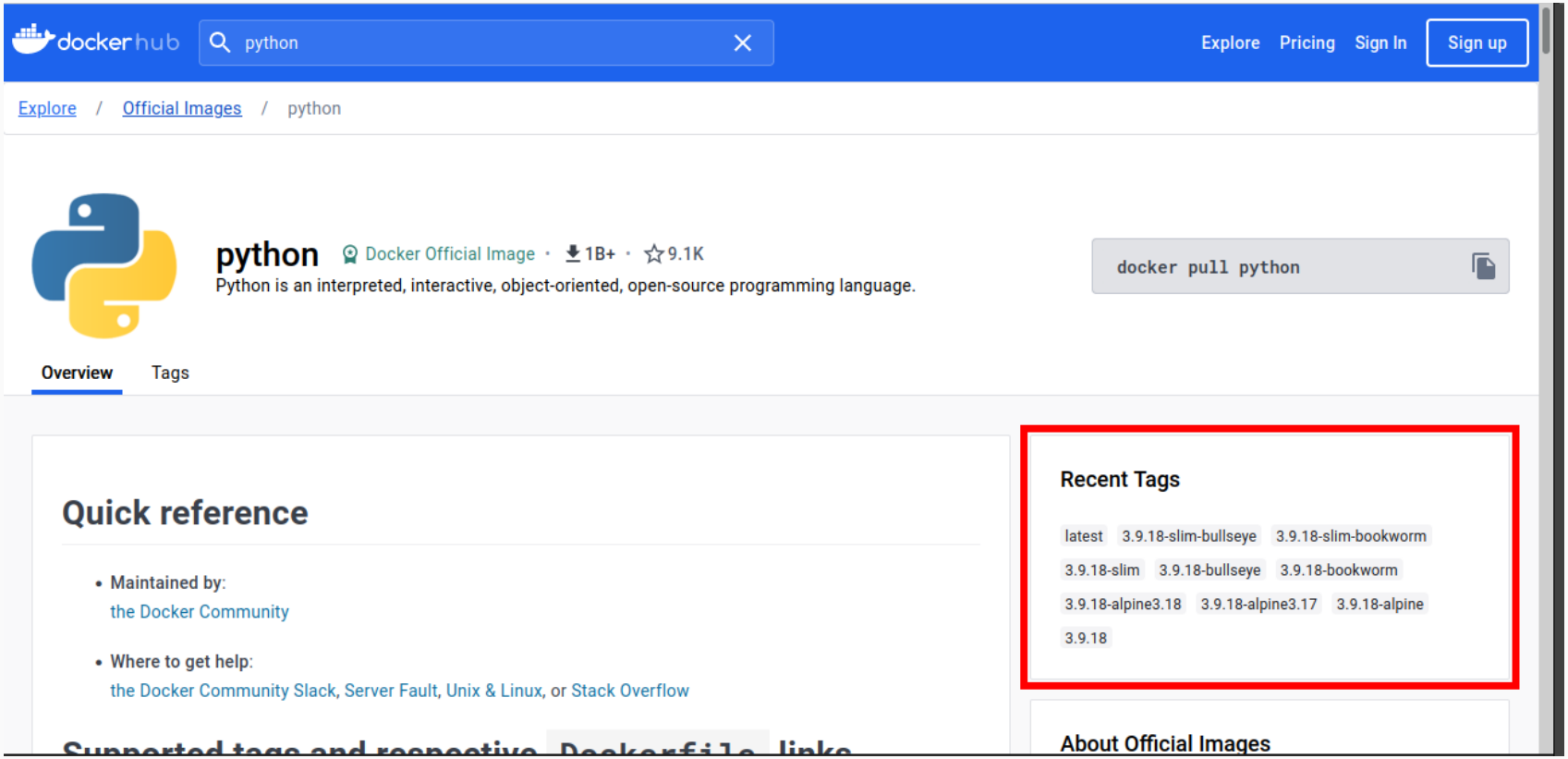
Dockerhub


dockerhub
python
Explore
Pricing
Sign In
Sign up
Filters
Products
Images
Extensions
Plugins
Trusted Content
Docker Official Image
Verified Publisher
Sponsored OSS
Operating Systems
Linux
Windows
Architectures
1 - 25 of 10,000 results for python.
Best Match
python
Docker Official Image
1B+
9.1K
Updated 13 days ago
Python is an interpreted, interactive, object-oriented, open-source programming language.
Windows Linux ARM 386 ARM 64 PowerPC 64 LE IBM Z mips64le x86-64
Pulls: 8,625,673
Last week
Learn more
pypy
Docker Official Image
10M+
379
Updated 6 days ago
PyPy is a fast, compliant alternative implementation of the Python language.
Linux Windows ARM x86-64 ARM 64 386 IBM Z PowerPC 64 LE
Pulls: 25,809
Last week
Learn more
hylang
Docker Official Image
10M+
58
Pulls: 2,879
Last week

Dockerhub


dockerhub
python
Explore
Pricing
Sign In
Sign up
Explore / Official Images / python
python
Docker Official Image · 1B+ · 9.1K
Python is an interpreted, interactive, object-oriented, open-source programming language.
docker pull python
Overview
Tags
Quick reference
Maintained by:
the Docker Community
Where to get help:
the Docker Community Slack, Server Fault, Unix & Linux, or Stack Overflow
Recent Tags
latest
3.9.18-slim-bullseye
3.9.18-slim-bookworm
3.9.18-slim
3.9.18-bullseye
3.9.18-bookworm
3.9.18-alpine3.18
3.9.18-alpine3.17
3.9.18-alpine
3.9.18
About Official Images

Dockerhub


dockerhub
python
Explore
Pricing
Sign In
Sign up
Explore / Official Images / python
python
Docker Official Image · 1B+ · 9.1K
Python is an interpreted, interactive, object-oriented, open-source programming language.
docker pull python
Overview
Tags
Quick reference
Maintained by:
the Docker Community
Where to get help:
the Docker Community Slack, Server Fault, Unix & Linux, or Stack Overflow
Recent Tags
latest
3.9.18-slim-bullseye
3.9.18-slim-bookworm
3.9.18-slim
3.9.18-bullseye
3.9.18-bookworm
3.9.18-alpine3.18
3.9.18-alpine3.17
3.9.18-alpine
3.9.18
About Official Images

Tags

Tags are a way to specify the version of a image. Example:

python:3.10.13-bookworm

python:3.9.18-alpine3.17

httpd:2.4-alpine3.18

postgresql:latest

Be careful, sometimes container break if the wrong image is used

Pulling, listing and deleting image

To pull an image

```
$docker pull alpine:3.18
```

List all pulled image

```
$docker image ls
```

Remove a previously pulled image

```
$docker image rm alpine:3.18
```

Start a container

General commandformat

```
$docker run [OPTIONS] IMAGE[:TAG|@DIGEST] \
    [COMMAND] [ARG...]
```

The image gets pulled if it does not exist.

Be careful, sometimes this is unwanted

```
$docker run hello-world
```

```
$docker run alpine:latest
```

Running container: interactive and tty flag

How do we interact with a shell in a Container?

--interactive [Doc23b]

Keep STDIN open even if not attached

--tty [Doc23b]

Allocate a pseudo-TTY

```
$docker run --interactive --tty alpine:3.18
```

Shorter and memorable example

```
$docker run -it alpine:3.18
```

Specify command

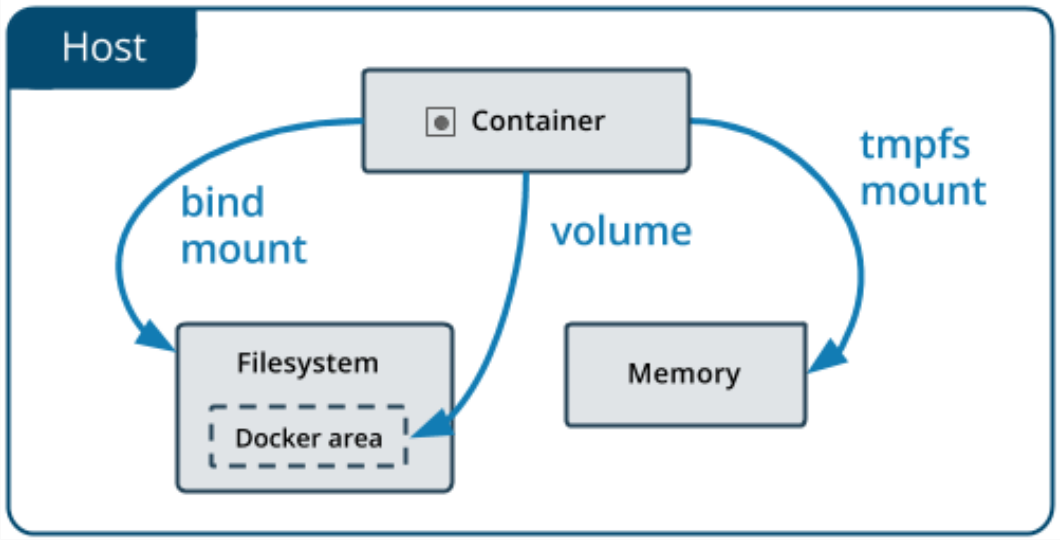
Run commands by appending the command to the docker run command

```
$docker run -i alpine:3.18 ls -la
total 64
drwxr-xr-x    1 root   root   4096 Nov  1 08:17 .
drwxr-xr-x    1 root   root   4096 Nov  1 08:17 ..
-rwxr-xr-x    1 root   root      0 Nov  1 08:17 .dockerenv
drwxr-xr-x    2 root   root   4096 Sep 28 11:18 bin
drwxr-xr-x    5 root   root    340 Nov  1 08:17 dev
drwxr-xr-x    1 root   root   4096 Nov  1 08:17 etc
drwxr-xr-x    2 root   root   4096 Sep 28 11:18 home
drwxr-xr-x    7 root   root   4096 Sep 28 11:18 lib
drwxr-xr-x    5 root   root   4096 Sep 28 11:18 media
drwxr-xr-x    2 root   root   4096 Sep 28 11:18 mnt
drwxr-xr-x    2 root   root   4096 Sep 28 11:18 opt
dr-xr-xr-x  336 root   root      0 Nov  1 08:17 proc
drwx------    2 root   root   4096 Sep 28 11:18 root
drwxr-xr-x    2 root   root   4096 Sep 28 11:18 run
drwxr-xr-x    2 root   root   4096 Sep 28 11:18 sbin
drwxr-xr-x    2 root   root   4096 Sep 28 11:18 srv
dr-xr-xr-x   13 root   root      0 Nov  1 08:17 sys
drwxrwxrwt    2 root   root   4096 Sep 28 11:18 tmp
drwxr-xr-x    7 root   root   4096 Sep 28 11:18 usr
drwxr-xr-x   12 root   root   4096 Sep 28 11:18 var
```

Mounttypes

Sometimes it might be useful to share files with the container
Per default Data get written to a writable container layer
There are the following options to share Data [Doc23d]

- bind mount: specify a directory
- volume mount: create a seperate volume
- tmpfs mount



bind mount

Create a bindmount and share the content in *tmp/example1* in the folder */example*

```
$tree example1/
example1/
---- dummy.txt

$docker run --mount \
    type=bind,src=/tmp/example1,dst=/example \
    alpine:3.18 ls example
```

Pitfalls: bind mount

Be careful!

- If a file gets changed on the host, these changes are visible to the container
- If a file gets changed in the container, these changes are visible on the host

Copy files from the container

Sometimes it might be usefull to copy data from the container to the hostmachien

```
docker cp [OPTIONS] CONTAINER:SRC_PATH DEST_PATH|-
```

Of from the hostmachine into the container

```
docker cp [OPTIONS] SRC_PATH|- CONTAINER:DEST_PATH
```

Expose Containerports

There are two ways of connecting to a service inside of a container

- Use the containers IP-Address
- expose the port to the Host

In the following example the Port 80 from the container is exposed as port 8080 on the hostmachine

```
$docker run -p 8080:80 python:3.10-alpine \
    python -m http.server 80
```

You should see the directory-listing from the container on `http://localhost:8080/`

docker ps

How do you list all running container?

```
$docker ps
```

Also list exited containers

```
$docker ps -a
```

The container names get randomized if no name is specified

```
$docker run -it --name alpine_318 alpine:3.18
```

```
CONTAINER ID   IMAGE          COMMAND    CREATED        STATUS        PORTS NAMES
6ef477612e35   alpine:3.18    "/bin/sh"  2 seconds ago  Up 2 seconds        alpine_318
```

Restart an exited container

Exited container are stored and can be restarted

```
$docker ps -a
```

```
CONTAINER ID   IMAGE        COMMAND   CREATED            STATUS                     NAMES
8f3c2f160dc2   hello-world  "/hello"  About an hour ago  Exited (0) 2 seconds ago   hello_01
```

They can be restarted using the ID or the name

```
$docker start -i hello_01
```

```
$docker start -i 8f3c2f160dc2
```

How to delete images and containers

Sometimes it might be useful to remove older images or container
Deleting images is possible with the following command[Doc23a]

```
$docker image rm <ID>
```

Alternatively there is a shorter version

```
$docker rmi <ID>
```

Deleting a container works the same

```
$docker container rm <ID/Name>
```

Housekeeping: pruning

Remove all unused resources to save space

Pruning[Doc23c]

Remove all unused containers, networks, images (both dangling and unreferenced), and optionally, volumes.

```
$docker system prune
```

Running container: rm flag

How do we throw away these small intermediate cotnainers?

--rm[Doc23b]

Automatically remove the container when it exits

Shorter and memorable example

```
$docker run --rm -it alpine:3.18
```

Other usefull arguments

--user [Doc23b]

Username or UID

--label [Doc23b]

Set meta data on a container

--network [Doc23b]

Connect a container to a network

--privileged[Doc23b]

Give extended privileges to this container

Privileged Containers

Running a container with *–privileged* exposes the *docker.socket* from the post inside of the container. This enables a container to spawn other containers

- This enables a container to spawn other containers

Don't do it for containers that you don't know

Inspecting Images

With the *inspect*-command we are able to get more information about the image

- What baseimage was used
- Are there any exposed ports
- What command is run once the container startes
- filesystem related information

```
$docker inspect <containername>
```

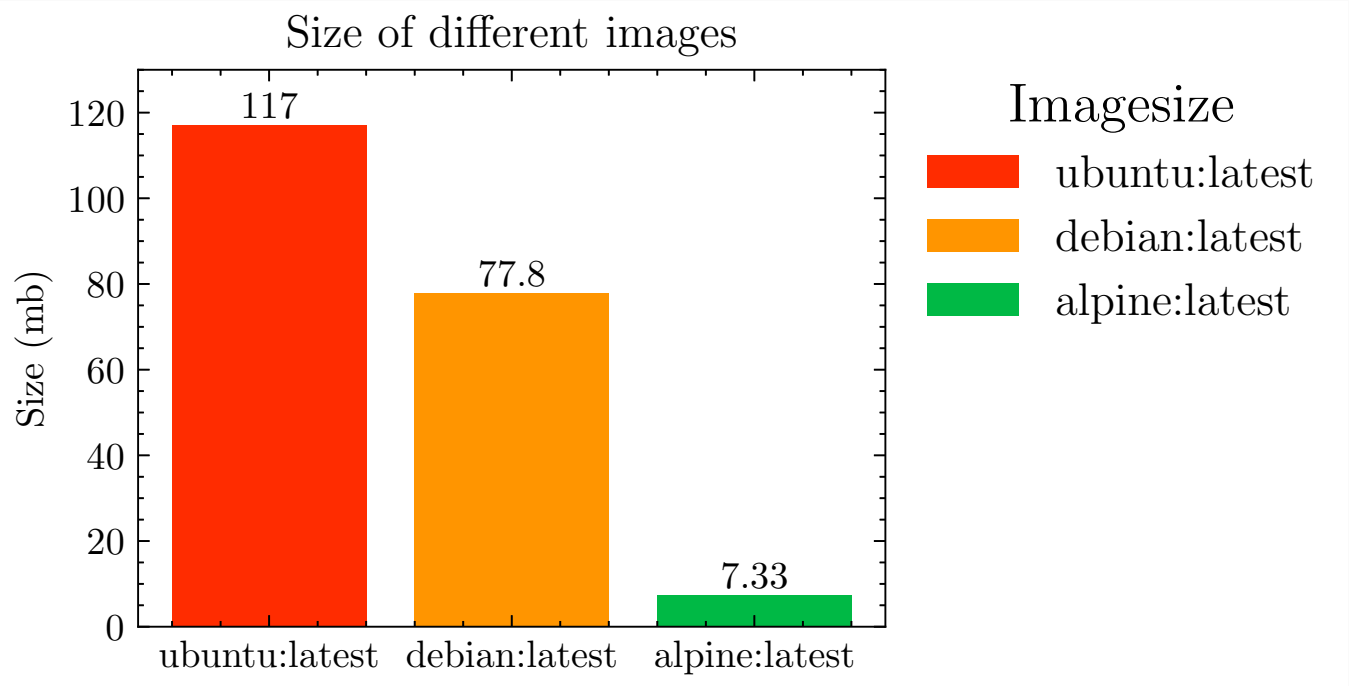
```
$docker inspect hello-world
```

Dockerfile

Alpine Linux

Alpine is a small, security focused distribution.

It gained popularity as baseimage for dockercontainer because of its small size



Alpine Linux: Drawbacks

- Alpine uses busybox (no bash)
- Alpine use musl as libc (not glibc)
- Software needs to be build to be linked against musl

Why is it important to choose stable tags

When coosing a tag, the image should build the same now and in 5 years (given the image and tag still exists then)

```
FROM debian:stable
RUN apt-get update
RUN apt-get install -y python3 python3-pip
COPY ./requirements.txt /app/requirements.txt
[....]
```

How to choose a tag

- Use a up to date version when writing the dockerfile
- Stick to well-known well-maintained baseimages
- Pin to a Major and a Minor Version

Instructions

Instruction	Purpose
FROM	Specify a baseimage
COPY	Copy files into the image
WORKDIR	Sets the working directory
RUN	Run a command in the container
ARG	Variable that can be set during build-time
CMD	Is run once the container is started.
EXPOSE	Exposes ports

Example Dockerfile

```
FROM python:3.10-alpine3.18
EXPOSE 8080
RUN pip install flask
COPY server.py /app/server.py
CMD python /app/server.py
```

What will be smaller

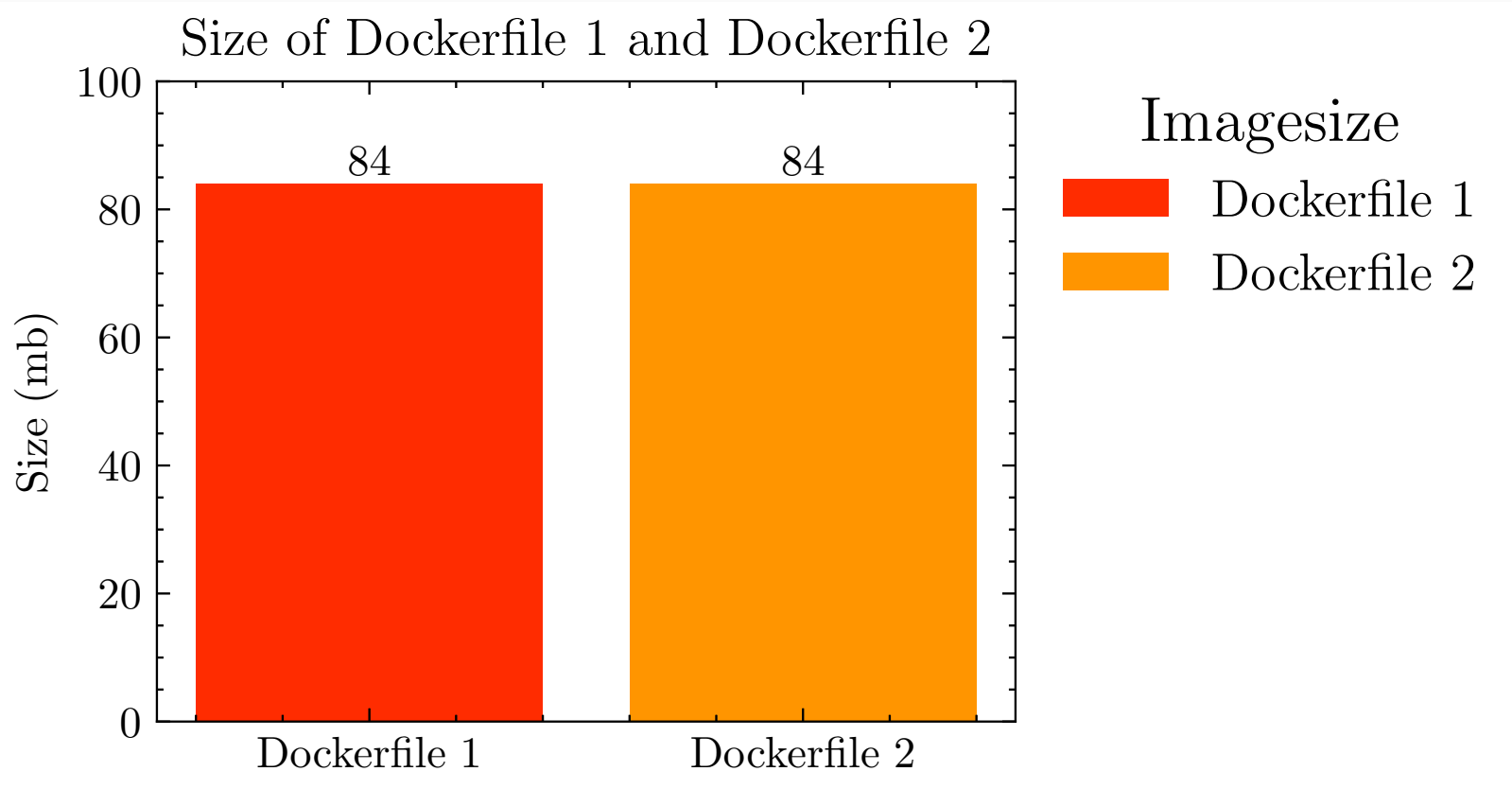
Dockerfile 1

```
FROM alpine:3.18
RUN apk upgrade
RUN apk add wget curl python3 py3-pip
```

Dockerfile 2

```
FROM alpine:3.18
RUN apk upgrade
RUN apk add wget curl python3 py3-pip
RUN apk del wget curl python3 py3-pip
```

What will be smaller: No one



What will be smaller: No one

```
docker inspect Dockerfile_1
```

```
"Layers": [
    "sha256:cc2447e1835a40530975ab80bb1f872fbab0f2a0faecf2ab16fbbb89b3589438",
    "sha256:10f18d1036ae31756f1448ed90fc44b873ce50ac1fdd6d76ac8b38318cad75d6",
    "sha256:fb825ad704e744d7433ae2052f97e09e6e9f428f06663614ae61153a8d0a0de7"
]
```

```
docker inspect Dockerfile_2
```

```
"Layers": [
    "sha256:cc2447e1835a40530975ab80bb1f872fbab0f2a0faecf2ab16fbbb89b3589438",
    "sha256:10f18d1036ae31756f1448ed90fc44b873ce50ac1fdd6d76ac8b38318cad75d6",
    "sha256:fb825ad704e744d7433ae2052f97e09e6e9f428f06663614ae61153a8d0a0de7",
    "sha256:57a303ce82c5229a310bb523309184dcf2e343e5312499d54e7fc488163009e9"
]
```

What will be smaller

Chain RUN-Comands

```
FROM alpine:3.18
RUN apk upgrade; \
    apk add wget curl python3 py3-pip; \
    apk del wget curl python3 py3-pip; \
```

Leads to the following filesize: *16.1MB*

What else is there?

- Docker Compose Stacks
- Networkmanagement
- Building a Dockercluster (Docker Swarm)
- ...

Try it yourselfe

What will be smaller: No one

We prepared some challenges for your.

They can all be solved with commands presented here

- Docker Runner 1
- Docker Runner 2
- Docker Inspector
- Docker Exposer
- Docker Copyer

Url: fuzzy.land

The name gives a hint on what to do

Also, *Docker Exposer* used information obtained during *Docker Inspector*

References i

[Doc23a] Docker. **Reference: docker rmi.** Nov. 2023. URL: https://docs.docker.com/engine/reference/commandline/rmi/.

[Doc23b] Docker. **Reference: docker run.** Nov. 2023. URL: https://docs.docker.com/engine/reference/commandline/run/.

[Doc23c] Docker. **Reference: docker system prune.** Nov. 2023. URL: https://docs.docker.com/engine/reference/commandline/system_prune/.

References ii

[Doc23d] Docker. **Reference: Manage data in Docker.** Nov. 2023. URL: https://docs.docker.com/storage/.

[Fer19] Simon Ferquel. **Docker 3 WSL 2: the future of docker desktop for windows.** June 2019. URL: https://www.docker.com/blog/docker-hearts-wsl-2/.