Intro to Docker

Containers & Images

elixir GREECE

ATHENA Research & Innovation Information Technologies

INAB INSTITUTE OF APPLIED BIOSCIENCES
ΙΝΣΤΙΤΟΥΤΟ ΕΦΑΡΜΟΣΜΕΝΩΝ ΒΙΟΕΠΙΣΤΗΜΩΝ
CENTRE for RECEARCH and TECHNOLOGY-HELLAS

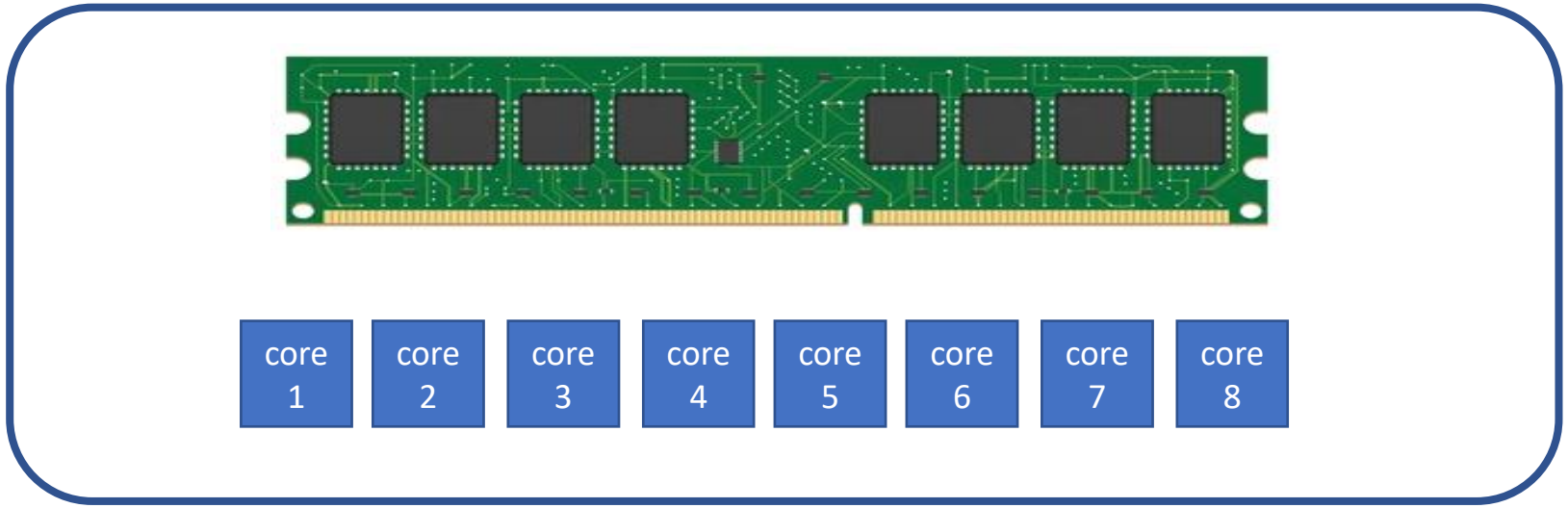
CERTH CENTRE FOR RESEARCH & TECHNOLOGY HELLAS

Virtual Machines (VMs)

- **Individual machines** inside a physical machine
- Use the resources of the main system (CPU, RAM, etc) via **virtualization technologies** (kvm, containerd, etc.)
- Have **their own** Operating System (OS) and software

Virtual Machines (VMs)

- **Individual machines** inside a physical machine
- Use the resources of the main system (CPU, RAM, etc) via **virtualization technologies** (kvm, containerd, etc.)
- Have **their own** Operating System (OS) and software

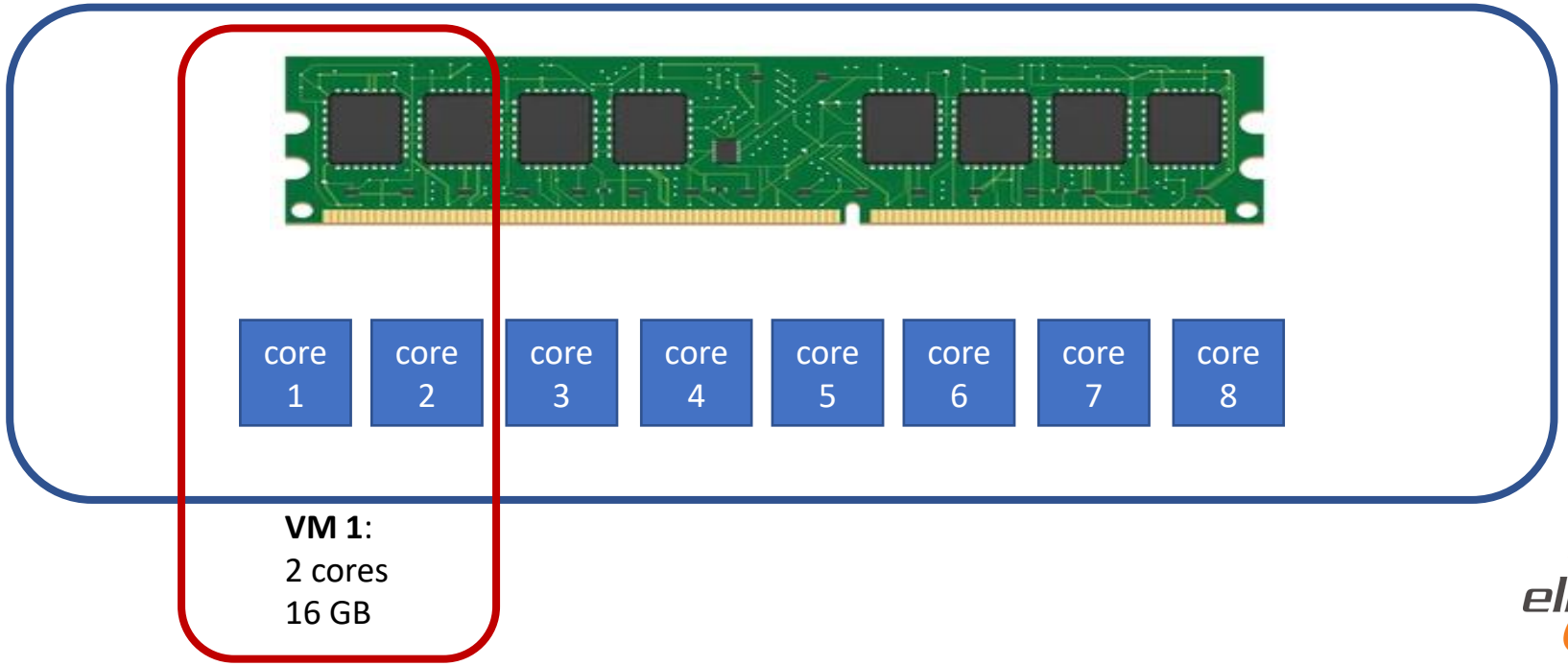


Physical Machine:

- 8 cpu cores
- 64 GB RAM

Virtual Machines (VMs)

- **Individual machines** inside a physical machine
- Use the resources of the main system (CPU, RAM, etc) via **virtualization technologies** (kvm, containerd, etc.)
- Have **their own** Operating System (OS) and software

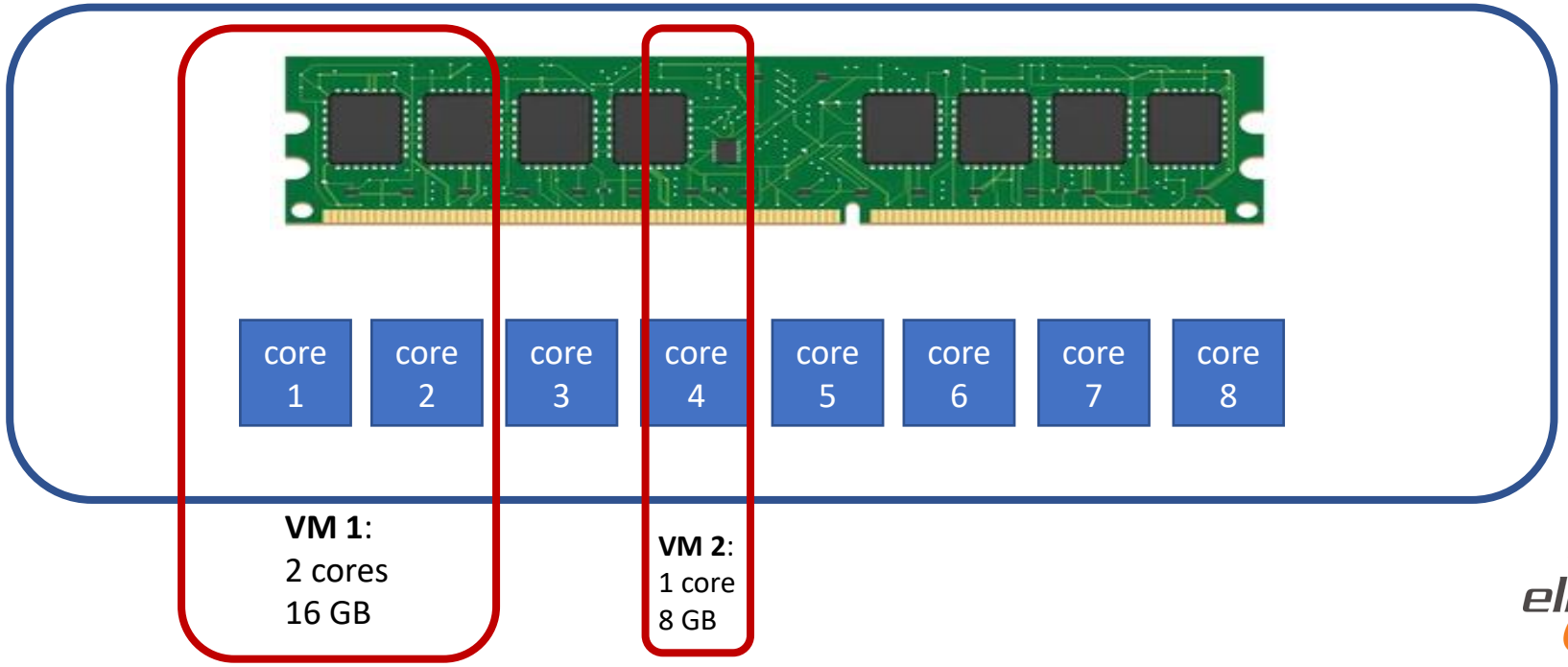


Physical Machine:

- 8 cpu cores
- 64 GB RAM

Virtual Machines (VMs)

- **Individual machines** inside a physical machine
- Use the resources of the main system (CPU, RAM, etc) via **virtualization technologies** (kvm, containerd, etc.)
- Have **their own** Operating System (OS) and software

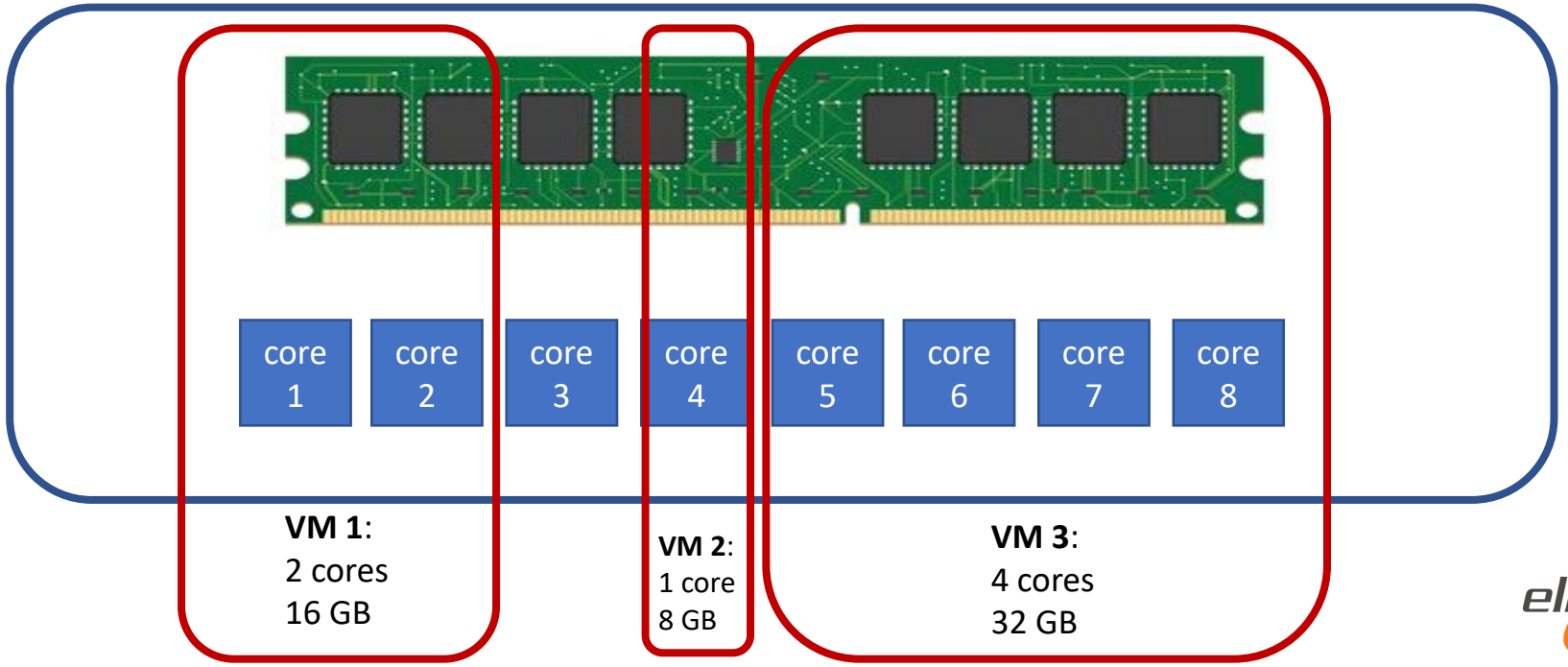


Physical Machine:

- 8 cpu cores
- 64 GB RAM

Virtual Machines (VMs)

- **Individual machines** inside a physical machine
- Use the resources of the main system (CPU, RAM, etc) via **virtualization technologies** (kvm, containerd, etc.)
- Have **their own** Operating System (OS) and software



Physical Machine:

- 8 cpu cores
- 64 GB RAM

Containers

- Virtual machines.
- Created using a specific **image**.
- Only exist to perform a **single task.**
- Die after task execution and resources freed.
- Elastic allocation of resources.

Container image

- Allows easy spawning of containers.
- Operating system + software required (user scripts, etc.)
- Container filesystem = image filesystem.
- Variety of images on public registries (mainly Docker Hub).

Docker

- Most commonly used software for containerization
- Based on containerd.
- Local image library
- https://docs.docker.com/

Creating containers

Containers & Images

Docker (command)

- Write “docker” on the command line.

Docker (command)

- Write “docker” on the command line.

```
ubuntu@cwl-docker-seminar-1:~$ docker

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Options:
      --config string      Location of client config files (default
                           "/home/ubuntu/.docker")
  -c, --context string     Name of the context to use to connect to the
                           daemon (overrides DOCKER_HOST env var and
                           default context set with "docker context use")
  -D, --debug              Enable debug mode
  -H, --host list          Daemon socket(s) to connect to
  -l, --log-level string   Set the logging level
                           ("debug"|"info"|"warn"|"error"|"fatal")
                           (default "info")
      --tls                Use TLS; implied by --tlsverify
      --tlscacert string   Trust certs signed only by this CA (default
                           "/home/ubuntu/.docker/ca.pem")
      --tlscert string     Path to TLS certificate file (default
                           "/home/ubuntu/.docker/cert.pem")
      --tlskey string      Path to TLS key file (default
```

Docker image

- **pull:** pull an image

Docker image

- **pull:** pull an image

```
ubuntu@cwl-docker-seminar-1:~$ docker image pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:37a0b92b08d4919615c3ee023f7ddb068d12b8387475d64c622ac30f45c29c51
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
```

Docker image

- **pull:** pull an image

```
ubuntu@cwl-docker-seminar-1:~$ docker image pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:37a0b92b08d4919615c3ee023f7ddb068d12b8387475d64c622ac30f45c29c51
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
```

- **ls:** show available images in the local repository

Docker image

- **pull:** pull an image

```
ubuntu@cwl-docker-seminar-1:~$ docker image pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:37a0b92b08d4919615c3ee023f7ddb068d12b8387475d64c622ac30f45c29c51
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
```

- **ls:** show available images in the local repository

```
ubuntu@cwl-docker-seminar-1:~$ docker image ls
REPOSITORY    TAG       IMAGE ID       CREATED       SIZE
hello-world   latest    feb5d9fea6a5   3 weeks ago   13.3kB
```

Docker image

- **pull:** pull an image

```
ubuntu@cwl-docker-seminar-1:~$ docker image pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:37a0b92b08d4919615c3ee023f7ddb068d12b8387475d64c622ac30f45c29c51
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
```

- **ls:** show available images in the local repository

```
ubuntu@cwl-docker-seminar-1:~$ docker image ls
REPOSITORY    TAG       IMAGE ID       CREATED       SIZE
hello-world   latest    feb5d9fea6a5   3 weeks ago   13.3kB
```

- **rm:** delete image

Docker image

- **pull:** pull an image

```
ubuntu@cwl-docker-seminar-1:~$ docker image pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:37a0b92b08d4919615c3ee023f7ddb068d12b8387475d64c622ac30f45c29c51
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
```

- **ls:** show available images in the local repository

```
ubuntu@cwl-docker-seminar-1:~$ docker image ls
REPOSITORY    TAG       IMAGE ID       CREATED       SIZE
hello-world   latest    feb5d9fea6a5   3 weeks ago   13.3kB
```

- **rm:** delete image

```
ubuntu@cwl-docker-seminar-1:~$ docker image rm hello-world:latest
Untagged: hello-world:latest
Untagged: hello-world@sha256:37a0b92b08d4919615c3ee023f7ddb068d12b8387475d64c622
ac30f45c29c51
Deleted: sha256:feb5d9fea6a5e9606aa995e879d862b825965ba48de054caab5ef356dc6b3412
Deleted: sha256:e07ee1baac5fae6a26f30cabfe54a36d3402f96afda318fe0a96cec4ca393359
```

Docker image (2)

- Image consists of two parts:

- **<name>:<tag>**
  - Name/repository
  - Tag/version

- **Examples:** ubuntu:latest, ubuntu:18.04, ubuntu:16.04, ubuntu:focal

Docker run

- Try “docker run --name <your username> hello-world”

Docker run

- Try “docker run hello-world”

```
ubuntu@cwl-docker-seminar-1:~$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:37a0b92b08d4919615c3ee023f7ddb068d12b8387475d64c622ac30f45c29c51
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.
```

Docker run (background)

- Try “docker run -d hello-world”

```
ubuntu@cwl-docker-seminar-1:~$ docker run -d hello-world
51a132cc4623f56562bf2dfa8c710f073af43fbd27c67303f7a95ad75a3fb85d
ubuntu@cwl-docker-seminar-1:~$
```

- To see the output of the container run: “docker logs <id>

```
ubuntu@cwl-docker-seminar-1:~$ docker logs 51a132cc4623f56562bf2dfa8c710f073af43
fbd27c67303f7a95ad75a3fb85d

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

Doker run (syntax)

- Usual docker syntax:
  docker run <arguments> <image:tag> <command>
  where:
  - **arguments**: can be one of -d, -r, --rm, -v, etc. (try docker run --help for a full list of arguments)
  - **image:tag:** the image to create a container
  - **command** (optional)**:** the command to run inside the container. *If empty, the container runs the command specified by the image creator.*
- Try “docker run --name <your username> ubuntu:focal”

Doker run (syntax)

- Usual docker syntax:
  docker run <arguments> <image:tag> <command>
  where:
  - **arguments**: can be one of -d, -r, --rm, -v, etc. (try docker run --help for a full list of arguments)
  - **image:tag:** the image to create a container
  - **command** (optional)**:** the command to run inside the container. *If empty, the container runs the command specified by the image creator.*
- Try “docker run --name <your username> ubuntu:focal”
- Try “docker run --name <your username> ubuntu:focal ls -lah /”

Doker run (syntax)

- Usual docker syntax:
  docker run <arguments> <image:tag> <command>
  where:
  - **arguments**: can be one of -d, -r, --rm, -v, etc. (try docker run --help for a full list of arguments)
  - **image:tag:** the image to create a container
  - **command** (optional)**:** the command to run inside the container. *If empty, the container runs the command specified by the image creator.*
- Try “docker run ubuntu:focal”
- Try “docker run ubuntu:focal ls -lah /”

```
ubuntu@cwl-docker-seminar-1:~$ docker run ubuntu:focal ls -lah /
total 56K
drwxr-xr-x   1 root root 4.0K Oct 19 13:22 .
drwxr-xr-x   1 root root 4.0K Oct 19 13:22 ..
-rwxr-xr-x   1 root root    0 Oct 19 13:22 .dockerenv
lrwxrwxrwx   1 root root    7 Oct  6 16:47 bin -> usr/bin
drwxr-xr-x   2 root root 4.0K Apr 15  2020 boot
drwxr-xr-x   5 root root  340 Oct 19 13:22 dev
drwxr-xr-x   1 root root 4.0K Oct 19 13:22 etc
drwxr-xr-x   2 root root 4.0K Apr 15  2020 home
lrwxrwxrwx   1 root root    7 Oct  6 16:47 lib -> usr/lib
lrwxrwxrwx   1 root root    9 Oct  6 16:47 lib32 -> usr/lib32
lrwxrwxrwx   1 root root    9 Oct  6 16:47 lib64 -> usr/lib64
```

Doker run (syntax)

- Usual docker syntax:
  docker run <arguments> <image:tag> <command>
  where:
  - **arguments**: can be one of -d, -r, --rm, -v, etc. (try docker run --help for a full list of arguments)
  - **image:tag:** the image to create a container
  - **command** (optional)**:** the command to run inside the container. *If empty, the container runs the command specified by the image creator.*
- Try “docker run ubuntu:focal”
- Try “docker run ubuntu:focal ls -lah /”
- Try “docker run -d ubuntu:focal sleep 3600”

```
ubuntu@cwl-docker-seminar-1:~$ docker run ubuntu:focal ls -lah /
total 56K
drwxr-xr-x   1 root root 4.0K Oct 19 13:22 .
drwxr-xr-x   1 root root 4.0K Oct 19 13:22 ..
-rwxr-xr-x   1 root root    0 Oct 19 13:22 .dockerenv
lrwxrwxrwx   1 root root    7 Oct  6 16:47 bin -> usr/bin
drwxr-xr-x   2 root root 4.0K Apr 15  2020 boot
drwxr-xr-x   5 root root  340 Oct 19 13:22 dev
drwxr-xr-x   1 root root 4.0K Oct 19 13:22 etc
drwxr-xr-x   2 root root 4.0K Apr 15  2020 home
lrwxrwxrwx   1 root root    7 Oct  6 16:47 lib -> usr/lib
lrwxrwxrwx   1 root root    9 Oct  6 16:47 lib32 -> usr/lib32
lrwxrwxrwx   1 root root    9 Oct  6 16:47 lib64 -> usr/lib64
```

Docker container

- **ls:** show active containers

Docker container

- **ls:** show active containers

```
ubuntu@cwl-docker-seminar-1:~$ docker container ls
CONTAINER ID   IMAGE          COMMAND        CREATED         STATUS         PORTS     NAMES
f9faef2219e0   ubuntu:focal   "sleep 3600"   4 seconds ago   Up 3 seconds             happy_swanson
```

Docker container

- **ls:** show active containers

```
ubuntu@cwl-docker-seminar-1:~$ docker container ls
CONTAINER ID   IMAGE          COMMAND        CREATED         STATUS         PORTS     NAMES
f9faef2219e0   ubuntu:focal   "sleep 3600"   4 seconds ago   Up 3 seconds             happy_swanson
```

- **stop:** stop active container

Docker container

- **ls:** show active containers

```
ubuntu@cwl-docker-seminar-1:~$ docker container ls
CONTAINER ID   IMAGE          COMMAND        CREATED         STATUS         PORTS     NAMES
f9faef2219e0   ubuntu:focal   "sleep 3600"   4 seconds ago   Up 3 seconds             happy_swanson
```

- **stop:** stop active container

```
ubuntu@cwl-docker-seminar-1:~$ docker stop happy_swanson
happy_swanson
```

Docker container

- ls: show active containers

```
ubuntu@cwl-docker-seminar-1:~$ docker container ls
CONTAINER ID   IMAGE          COMMAND        CREATED         STATUS         PORTS     NAMES
f9faef2219e0   ubuntu:focal   "sleep 3600"   4 seconds ago   Up 3 seconds             happy_swanson
```

- stop: stop active container

```
ubuntu@cwl-docker-seminar-1:~$ docker stop happy_swanson
happy_swanson
```

- start: start active container (after it is stopped)

Docker container

- **ls:** show active containers

```
ubuntu@cwl-docker-seminar-1:~$ docker container ls
CONTAINER ID   IMAGE          COMMAND        CREATED         STATUS         PORTS     NAMES
f9faef2219e0   ubuntu:focal   "sleep 3600"   4 seconds ago   Up 3 seconds             happy_swanson
```

- **stop:** stop active container

```
ubuntu@cwl-docker-seminar-1:~$ docker stop happy_swanson
happy_swanson
```

- **start:** start active container (after it is stopped)

```
ubuntu@cwl-docker-seminar-1:~$ docker start happy_swanson
happy_swanson
```

Docker container

- **ls:** show active containers

```
ubuntu@cwl-docker-seminar-1:~$ docker container ls
CONTAINER ID   IMAGE          COMMAND        CREATED         STATUS         PORTS     NAMES
f9faef2219e0   ubuntu:focal   "sleep 3600"   4 seconds ago   Up 3 seconds             happy_swanson
```

- **stop:** stop active container

```
ubuntu@cwl-docker-seminar-1:~$ docker stop happy_swanson
happy_swanson
```

- **start:** start active container (after it is stopped)

```
ubuntu@cwl-docker-seminar-1:~$ docker start happy_swanson
happy_swanson
```

- **restart:**

Docker container

- **ls:** show active containers

```
ubuntu@cwl-docker-seminar-1:~$ docker container ls
CONTAINER ID   IMAGE          COMMAND        CREATED         STATUS         PORTS     NAMES
f9faef2219e0   ubuntu:focal   "sleep 3600"   4 seconds ago   Up 3 seconds             happy_swanson
```

- **stop:** stop active container

```
ubuntu@cwl-docker-seminar-1:~$ docker stop happy_swanson
happy_swanson
```

- **start:** start active container (after it is stopped)

```
ubuntu@cwl-docker-seminar-1:~$ docker start happy_swanson
happy_swanson
```

- **restart:**

```
ubuntu@cwl-docker-seminar-1:~$ docker restart happy_swanson
happy_swanson
```

Docker container (2)

- **rm:**

Docker container (2)

- **rm:**

```
ubuntu@cwl-docker-seminar-1:~$ docker rm happy_swanson
Error response from daemon: You cannot remove a running container f9faef2219e077b63cbaa5053984049c431c77a068b187134f1c6de26197e576. Stop the container before attempting removal or force remove
```

Docker container (2)

- **rm:**

```
ubuntu@cwl-docker-seminar-1:~$ docker rm happy_swanson
Error response from daemon: You cannot remove a running container f9faef2219e077b63cbaa5053984049c431c77a068b187134f1c6de26197e576. Stop the container before attempting removal or force remove
```

- **rm -f:**

Docker container (2)

- **rm:**

```
ubuntu@cwl-docker-seminar-1:~$ docker rm happy_swanson
Error response from daemon: You cannot remove a running container f9faef2219e077b63cbaa5053984049c431c77a068b187134f1c6de26197e576. Stop the container before attempting removal or force remove
```

- **rm -f:**

```
ubuntu@cwl-docker-seminar-1:~$ docker rm -f happy_swanson
happy_swanson
```

Docker container (2)

- **rm:**

```
ubuntu@cwl-docker-seminar-1:~$ docker rm happy_swanson
Error response from daemon: You cannot remove a running container f9faef2219e077b63cbaa5053984049c431c77a068b187134f1c6de26197e576. Stop the container before attempting removal or force remove
```

- **rm -f:**

```
ubuntu@cwl-docker-seminar-1:~$ docker rm -f happy_swanson
happy_swanson
```

- Automatically delete container after it is stopped:
  docker run -d ***--rm*** *ubuntu:latest sleep 3600*

Docker container (2)

- **rm:**

```
ubuntu@cwl-docker-seminar-1:~$ docker rm happy_swanson
Error response from daemon: You cannot remove a running container f9faef2219e077b63cbaa5053984049c431c77a068b187134f1c6de26197e576. Stop the container before attempting removal or force remove
```

- **rm -f:**

```
ubuntu@cwl-docker-seminar-1:~$ docker rm -f happy_swanson
happy_swanson
```

- Automatically delete container after it is stopped:
  docker run -d ***--rm*** *ubuntu:latest sleep 3600*

```
ubuntu@cwl-docker-seminar-1:~$ docker run -d --rm ubuntu:latest sleep 3600
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
Digest: sha256:626ffe58f6e7566e00254b638eb7e0f3b11d4da9675088f4781a50ae288f3322
Status: Downloaded newer image for ubuntu:latest
786f7cdb1274801dbaa9df813643bdb8418ec94110e33d7c11f2dbedd9beab4d
ubuntu@cwl-docker-seminar-1:~$ docker container ls
CONTAINER ID   IMAGE          COMMAND        CREATED          STATUS          P
ORTS     NAMES
786f7cdb1274   ubuntu:latest   "sleep 3600"   29 seconds ago   Up 28 seconds
         vigilant_fermi
ubuntu@cwl-docker-seminar-1:~$ docker container stop vigilant_fermi
vigilant_fermi
ubuntu@cwl-docker-seminar-1:~$ docker container ls
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
ubuntu@cwl-docker-seminar-1:~$
```

Docker

- **ps –a:** list all processes (container ls --all)

```
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
ubuntu@cwl-docker-seminar-1:~$ docker ps -a
CONTAINER ID   IMAGE              COMMAND              CREATED             STATUS                          PORTS     NAMES
74516303cbfa   ubuntu:focal       "ls -lah /"          53 minutes ago      Exited (0) 53 minutes ago                 lucid_wu
bd4de43ddd10   ubuntu:focal       "sleep ls -lah /"    53 minutes ago      Exited (1) 53 minutes ago                 pedantic_brown
6de2782c4a9c   ubuntu:focal       "bash"               55 minutes ago      Exited (0) 55 minutes ago                 jolly_antonelli
7c8bac1d26ae   hello-world:linux  "cd /"               55 minutes ago      Created                                   hungry_euler
c3a4b16ed5c4   hello-world:linux  "ls -lah /"          56 minutes ago      Created                                   adoring_elgamal
51a132cc4623   hello-world        "/hello"             About an hour ago   Exited (0) About an hour ago              objective_jang
1ff84deb4d5f   hello-world        "/hello"             About an hour ago   Exited (0) About an hour ago              gallant_mccarthy
```

- for line in $(docker ps -a | tail -n+2 | awk '{print $1}'); do docker container rm -f $line; done;

```
ubuntu@cwl-docker-seminar-1:~$ for line in $(docker ps -a | tail  -n+2 | rev | tr -s ' ' | cut -d $' ' -f 1 | rev); do docker container rm -f $line; done;
lucid_wu
pedantic_brown
jolly_antonelli
hungry_euler
adoring_elgamal
objective_jang
gallant_mccarthy
```

Going into active container (exec)

- Run “docker run -d --rm ubuntu:latest sleep 3600”.
- Run ”docker container ls” to get the name or id of the container.
- Run “docker exec -it <container-name-or-id> /bin/bash”.
- You can now run commands inside the container.

Going into active container (exec)

- Run “docker run -d --rm ubuntu:latest sleep 3600”.
- Run ”docker container ls” to get the name or id of the container.
- Run “docker exec -it <container-name-or-id> /bin/bash”.
- You can now run commands inside the container.

```
ubuntu@cwl-docker-seminar-1:~$ docker exec -it trusting_blackburn /bin/bash
root@1732ac699fb6:/# ls /home
root@1732ac699fb6:/# ls /
bin   dev  home  lib32  libx32  mnt  proc  run   srv  tmp  var
boot  etc  lib   lib64  media   opt  root  sbin  sys  usr
root@1732ac699fb6:/#
```

Exposing container

- Why:
  - Web server
  - Jupyter server
  - User interaction over the internet

Exposing container

- Why:
  - Web server
  - Jupyter server
  - User interaction over the internet

Exposing container

- Why:
  - Web server
  - Jupyter server
  - User interaction over the internet

Exposing container

- Why:
  - Web server
  - Jupyter server
  - User interaction over the internet

Exposing container

- Why:
  - Web server
  - Jupyter server
  - User interaction over the internet

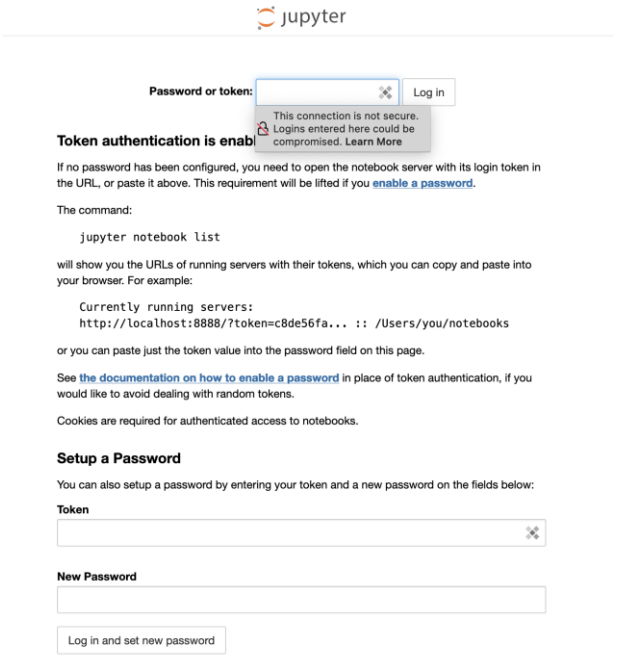
docker run -p 8080:8888 jupyter/datascience-notebook

Exposing container

- Why:
  - Web server
  - Jupyter server
  - User interaction over the internet

docker run -p 8080:8888 jupyter/datascience-notebook

http://machine-ip:8080



Data persistence

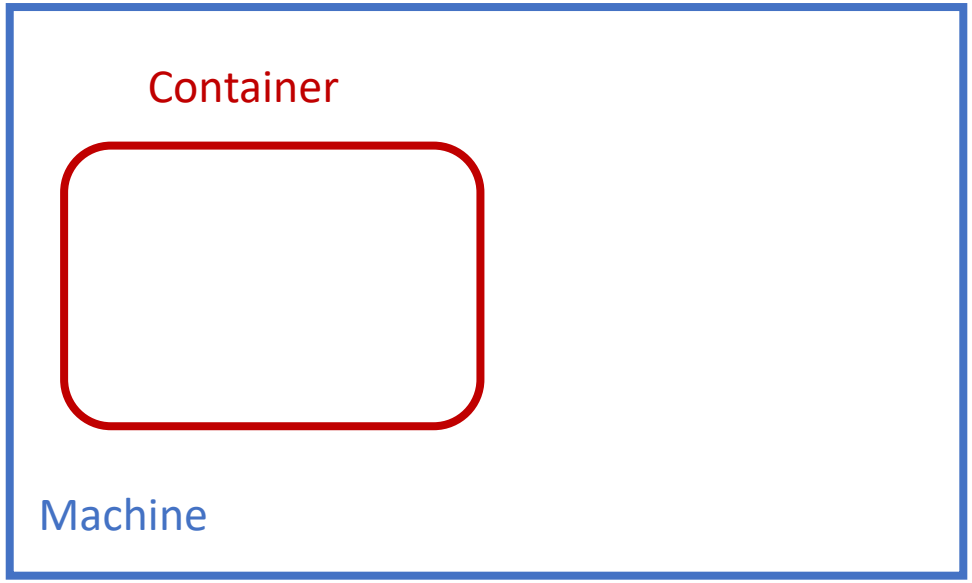
- What about files written inside a container?
  - *Deleted after container is destroyed!*

- How to save files in my local folder?

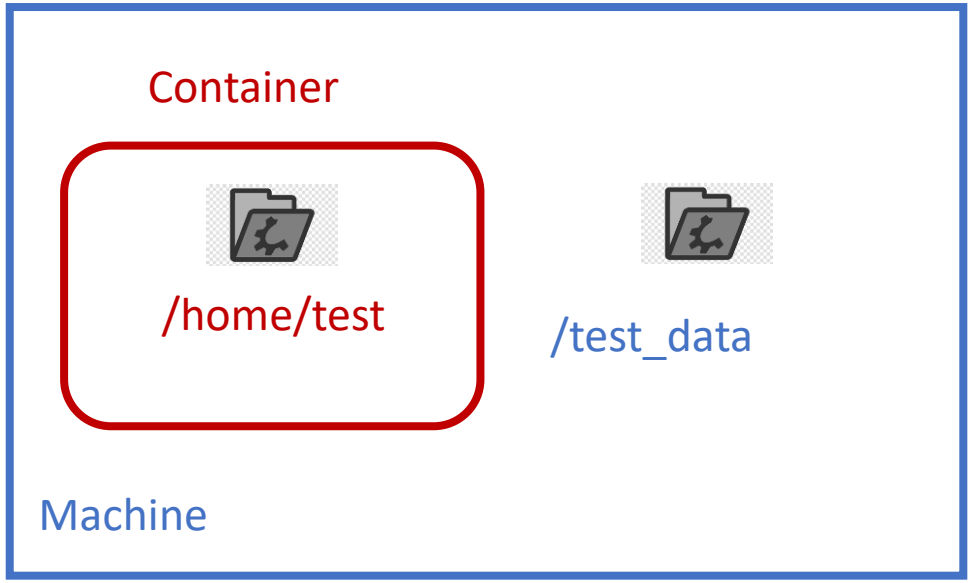
- How to pass files from a local folder to the container?

Mount local folder to the container!

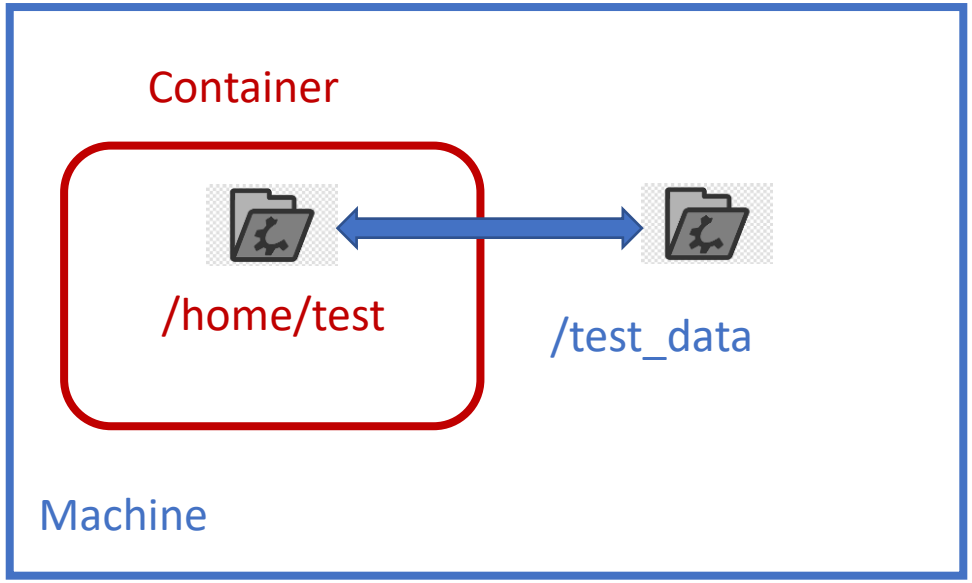
Data persistence



Data persistence

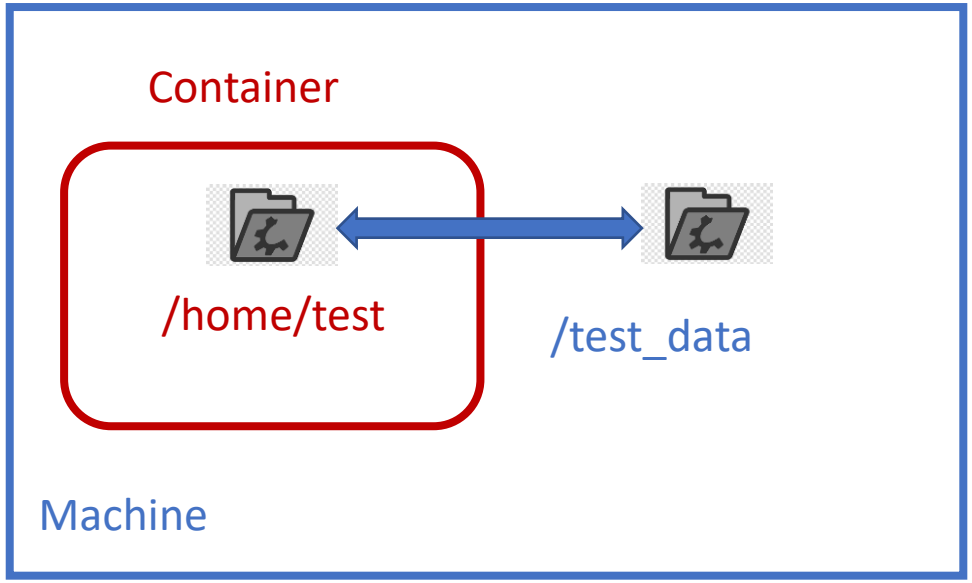


Data persistence



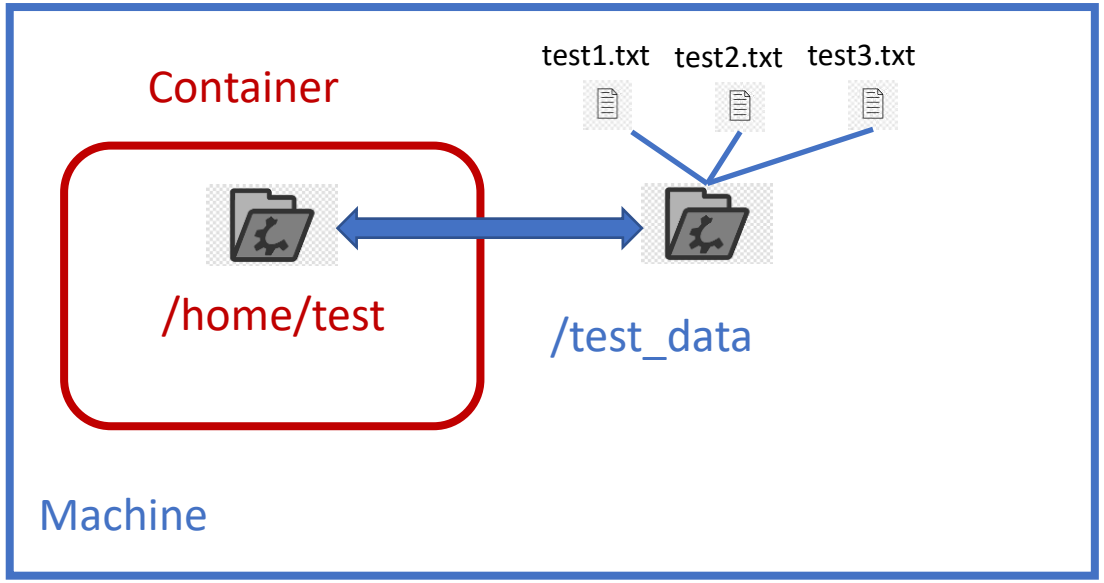
Data persistence

docker run -v <local-folder>:<cont-mountpoint> <container>



Data persistence

docker run -v <local-folder>:<cont-mountpoint> <container>



mkdir -p test_data

for i in $(seq 1 3); do touch test_data/test${i}.txt; done

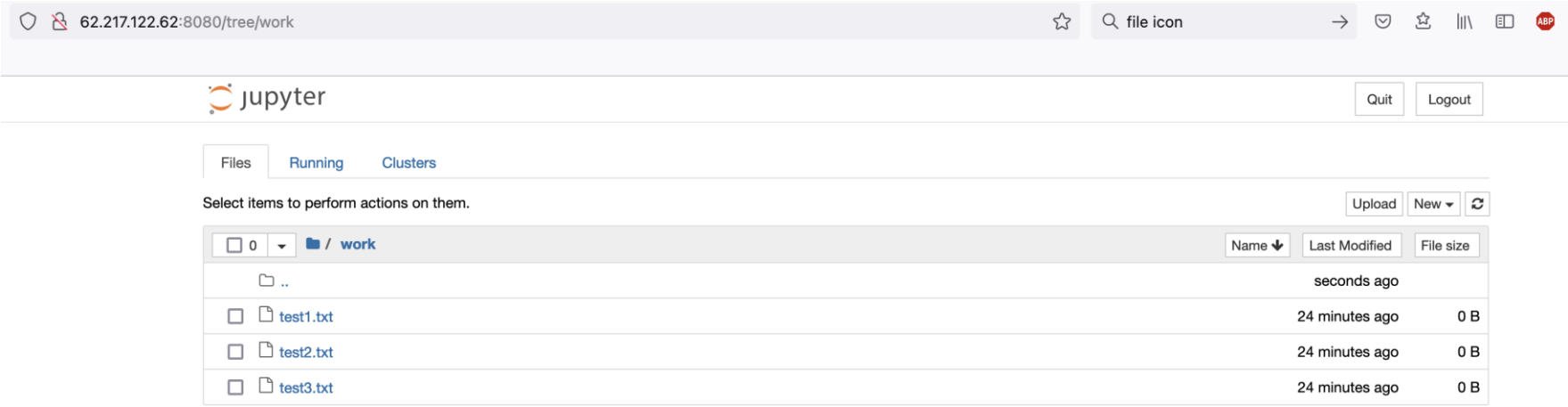
docker run -d -v ${PWD}/test_data:/home/test --rm ubuntu:focal sleep 3600

```
(base) kkyritsis@DESKTOP-LMCBDG2:~$ docker ps
CONTAINER ID   IMAGE          COMMAND        CREATED              STATUS              PORTS     NAMES
9ffb1b3edd3b   ubuntu:focal   "sleep 3600"   About a minute ago   Up About a minute             crazy_dubinsky
(base) kkyritsis@DESKTOP-LMCBDG2:~$ docker exec -it crazy_dubinsky /bin/bash
root@9ffb1b3edd3b:/# ls /home/test
test1.txt  test2.txt  test3.txt
root@9ffb1b3edd3b:/#
```

Data persistence

- Run a jupyter notebook mounted that saves notebooks inside a local folder:

```
docker run -p some_external_port:8888 \
-v <local_folder>:/home/jovyan/work \
--rm jupyter/datascience-notebook
```



Environment variables

- Add environment variables with:
  -e <VAR_NAME>=“<value>”

Environment variables

- Add environment variables with:
  -e <VAR_NAME>=“<value>”

- *docker run -d -v /test_data:/home/test *
  -e TEST_VAR="something" --rm ubuntu:focal sleep 3600

Environment variables

- Add environment variables with:
  -e <VAR_NAME>=“<value>”

- *docker run -d -v /test_data:/home/test *
  -e TEST_VAR="something" --rm ubuntu:focal sleep 3600

```
(base) kkyritsis@DESKTOP-LMCBDG2:~$ docker ps
CONTAINER ID   IMAGE          COMMAND        CREATED          STATUS          PORTS     NAMES
e51af2877f4f   ubuntu:focal   "sleep 3600"   19 seconds ago   Up 18 seconds             musing_thompson
9ffb1b3edd3b   ubuntu:focal   "sleep 3600"   4 minutes ago    Up 4 minutes              crazy_dubinsky
(base) kkyritsis@DESKTOP-LMCBDG2:~$ docker exec -it musing_thompson /bin/bash
root@e51af2877f4f:/# echo $TEST_VAR
something
root@e51af2877f4f:/#
```

Building images

elixir GREECE

ATHENA Research & Innovation Information Technologies

INAB INSTITUTE OF APPLIED BIOSCIENCES
ΙΝΣΤΙΤΟΥΤΟ ΕΦΑΡΜΟΣΜΕΝΩΝ ΒΙΟΕΠΙΣΤΗΜΩΝ
CENTRE for RECEARCH and TECHNOLOGY-HELLAS

CERTH CENTRE FOR RESEARCH & TECHNOLOGY HELLAS

Docker build

- Builds an image based on a Dockerfile
- Dockerfile: file containing instructions on how to create an image
- Most commonly used instructions:
  - FROM
  - RUN
  - COPY
  - CMD
  - WORKDIR

Dockerfile

Dockerfile

```
FROM ubuntu:focal

RUN apt update && apt install -y python3

RUN mkdir /home/test

COPY ./hello.py /home/test/

WORKDIR /home/test

CMD ["python3","hello.py"]

~
~
~
~
~
```

hello.py

```
print("Hello from a Docker container!")
```

Build the image: *docker build --file hello.dockerfile -t hello_py:1.0 .*

```
(base) kkyritsis@DESKTOP-LMCBDG2:~$ docker run --rm hello_py:1.0
Hello from a Docker container!
```

Dockerfile (in a nutshell)

- FROM: use existing image as a base
- RUN: run any command (used to install libraries, create filesystem structure, etc.)
- COPY: copy files/directories from a local folder to the image
- CMD: command to be executed with docker run (prefer array of strings)
- WORKDIR: change the working directory

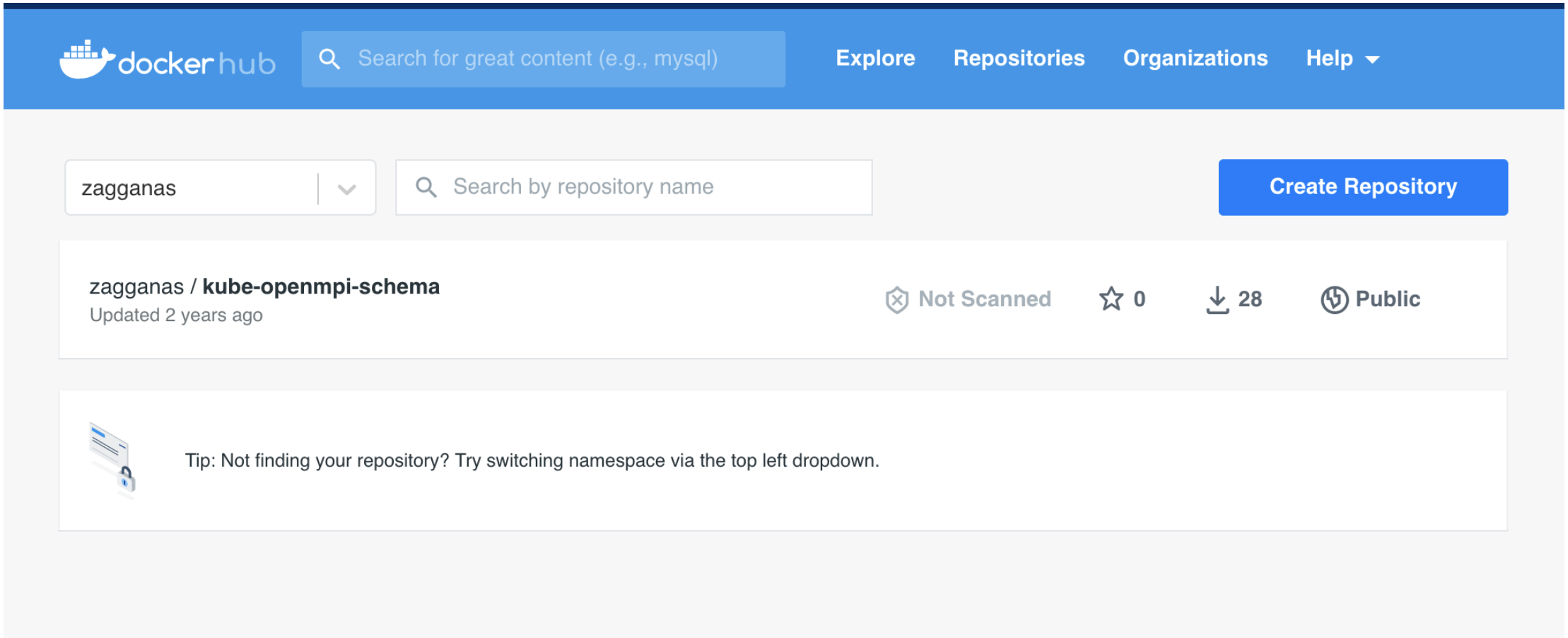
Docker Registries

- Host images
- Allow image pulling
- Allow image pushing
- Most popular registry: Docker Hub (https://hub.docker.com/)

Pushing to Docker Hub

- Create an account on https://hub.docker.com/
- Login to Docker Hub via command line.
- Build an image and tag it accordingly
- Push the image

Creating Docker Hub account



Pushing to Docker Hub

- Create an account on https://hub.docker.com/
- Login to Docker Hub via command line.
- Build an image and tag it accordingly
- Push the image

Login to Docker Hub

```
(base) kkyritsis@DESKTOP-LMCBDG2:~$ docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://h
ub.docker.com to create one.
Username: konstantinoskyritsis
Password:
Login Succeeded
```

Pushing to Docker Hub

- Create an account on https://hub.docker.com/
- Login to Docker Hub via command line.
- Build an image and tag it accordingly
- Push the image

Build and tag image

```
(base) kkyritsis@DESKTOP-LMCBDG2:/mnt/e/CERTH/WES_tutorial/Docker_CWL_tutorial$ docker build --file hello.dockerfile -t
konstantinoskyritsis/hello_py:1.0 .
[+] Building 16.8s (11/11) FINISHED
 => [internal] load build definition from hello.dockerfile                                                    0.0s
 => => transferring dockerfile: 44B                                                                           0.0s
 => [internal] load .dockerignore                                                                             0.0s
 => => transferring context: 2B                                                                               0.0s
 => [internal] load metadata for docker.io/library/ubuntu:focal                                              16.6s
 => [auth] library/ubuntu:pull token for registry-1.docker.io                                                 0.0s
 => [1/5] FROM docker.io/library/ubuntu:focal@sha256:450e066588f42ebe1551f3b1a535034b6aa46cd936fe7f2c6b0d72997ec6  0.0s
 => => resolve docker.io/library/ubuntu:focal@sha256:450e066588f42ebe1551f3b1a535034b6aa46cd936fe7f2c6b0d72997ec6  0.0s
 => [internal] load build context                                                                             0.0s
 => => transferring context: 29B                                                                              0.0s
 => CACHED [2/5] RUN apt update && apt install -y python3                                                     0.0s
 => CACHED [3/5] RUN mkdir /home/test                                                                         0.0s
 => CACHED [4/5] COPY ./hello.py /home/test                                                                   0.0s
 => CACHED [5/5] WORKDIR /home/test                                                                           0.0s
 => exporting to image                                                                                        0.0s
 => => exporting layers                                                                                       0.0s
 => => writing image sha256:6a031cc86718023f7fcfc5c1cd4bf9f3c348f88d97145186afaf155ef56ded93                  0.0s
 => => naming to docker.io/konstantinoskyritsis/hello_py:1.0                                                  0.0s
```

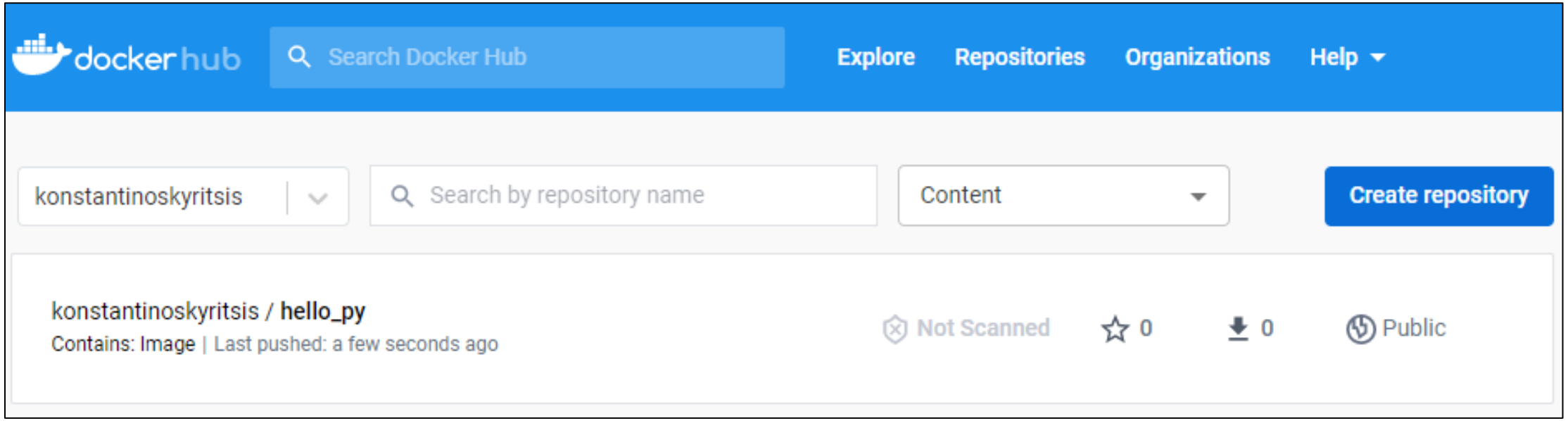
Pushing to Docker Hub

- Create an account on https://hub.docker.com/
- Login to Docker Hub via command line.
- Build an image and tag it accordingly
- Push the image

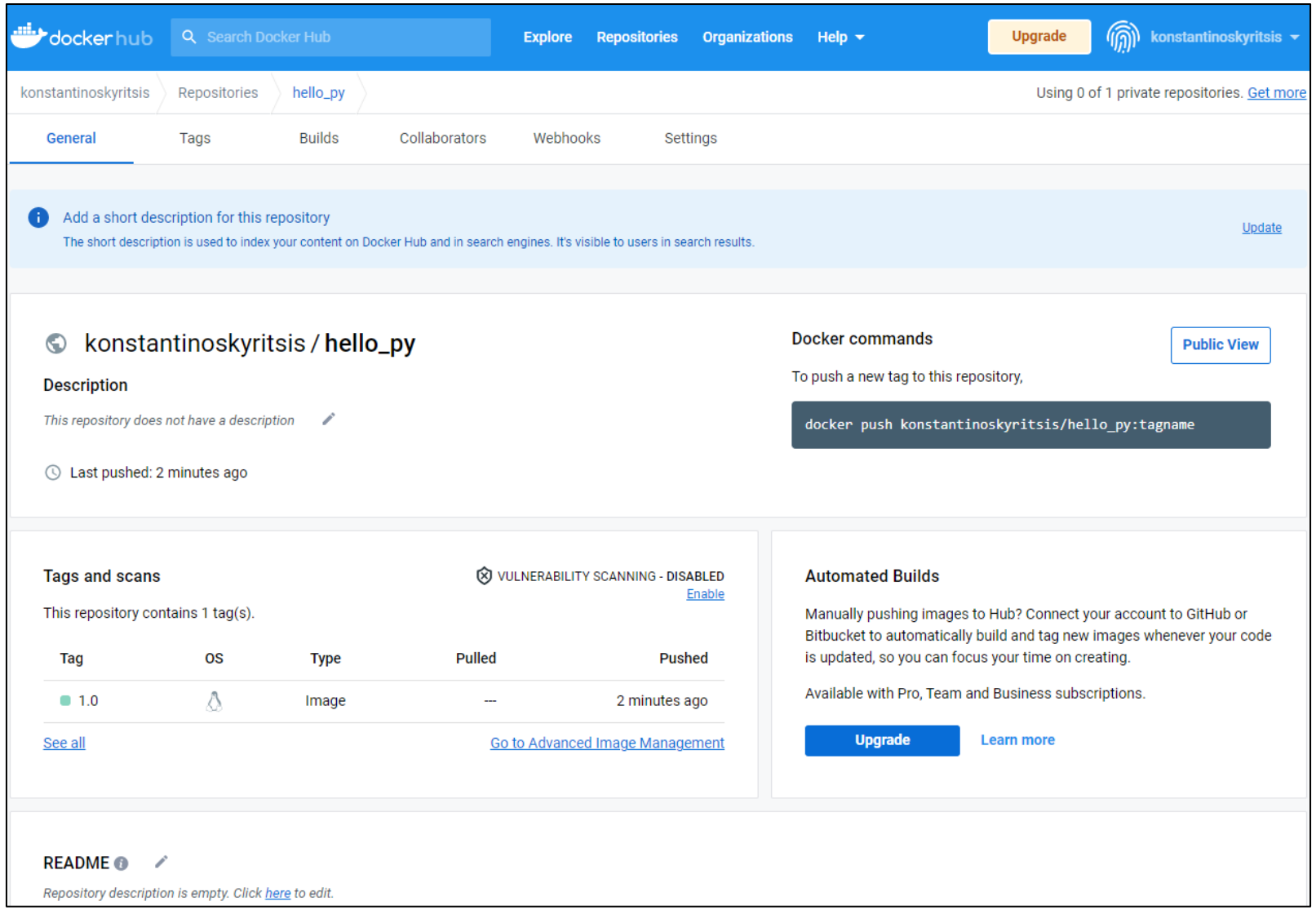
Push

```
(base) kkyritsis@DESKTOP-LMCBDG2:/mnt/e/CERTH/WES_tutorial/Docker_CWL_tutorial$ docker push konstantinoskyritsis/hello_p
y:1.0
The push refers to repository [docker.io/konstantinoskyritsis/hello_py]
5f70bf18a086: Pushed
e74556faa215: Pushed
10200e0c0998: Pushed
7f8fbc2158e0: Pushed
f4462d5b2da2: Mounted from library/ubuntu
1.0: digest: sha256:132518ddcce30f43ce7128f355cc30b7c0a00cf333e96962209c5f3234219fe1 size: 1361
```

Notice the new repo that appeared



Repo details



Pull and try image

```
(base) kkyritsis@DESKTOP-LMCBDG2:/mnt/e/CERTH/WES_tutorial/Docker_CWL_tutorial$ docker run --rm konstantinoskyritsis/hel
lo_py:1.0
Unable to find image 'konstantinoskyritsis/hello_py:1.0' locally
1.0: Pulling from konstantinoskyritsis/hello_py
eaead16dc43b: Already exists
aae6bb851cc3: Already exists
f8edb22f0271: Already exists
7baad86dbd57: Already exists
4f4fb700ef54: Already exists
Digest: sha256:132518ddcce30f43ce7128f355cc30b7c0a00cf333e96962209c5f3234219fe1
Status: Downloaded newer image for konstantinoskyritsis/hello_py:1.0
Hello from a Docker container!
```