



## SOFTWARE ARCHITECTURE

2023-24

Jose Emilio Labra Gayo  
Pablo González  
Cristian Augusto Alonso  
Jorge Álvarez Fidalgo

### Lab 3

## Deployment concepts



## Deployment

Deploying an application requires:

- Compile source code
- Get dependencies
- Configure environment
- Packaging
- Send to host machine
- Launch in execution environment

# Execution environment

- Where will the software be run?
- What dependencies does it have?
  - Operating system
  - Shared libraries
- Several options
  - Physical machines
  - Virtual machines
  - Containers



# Several ways to do the deployment

## Manually

It can be easier initially when there are few deployments

## Automatic

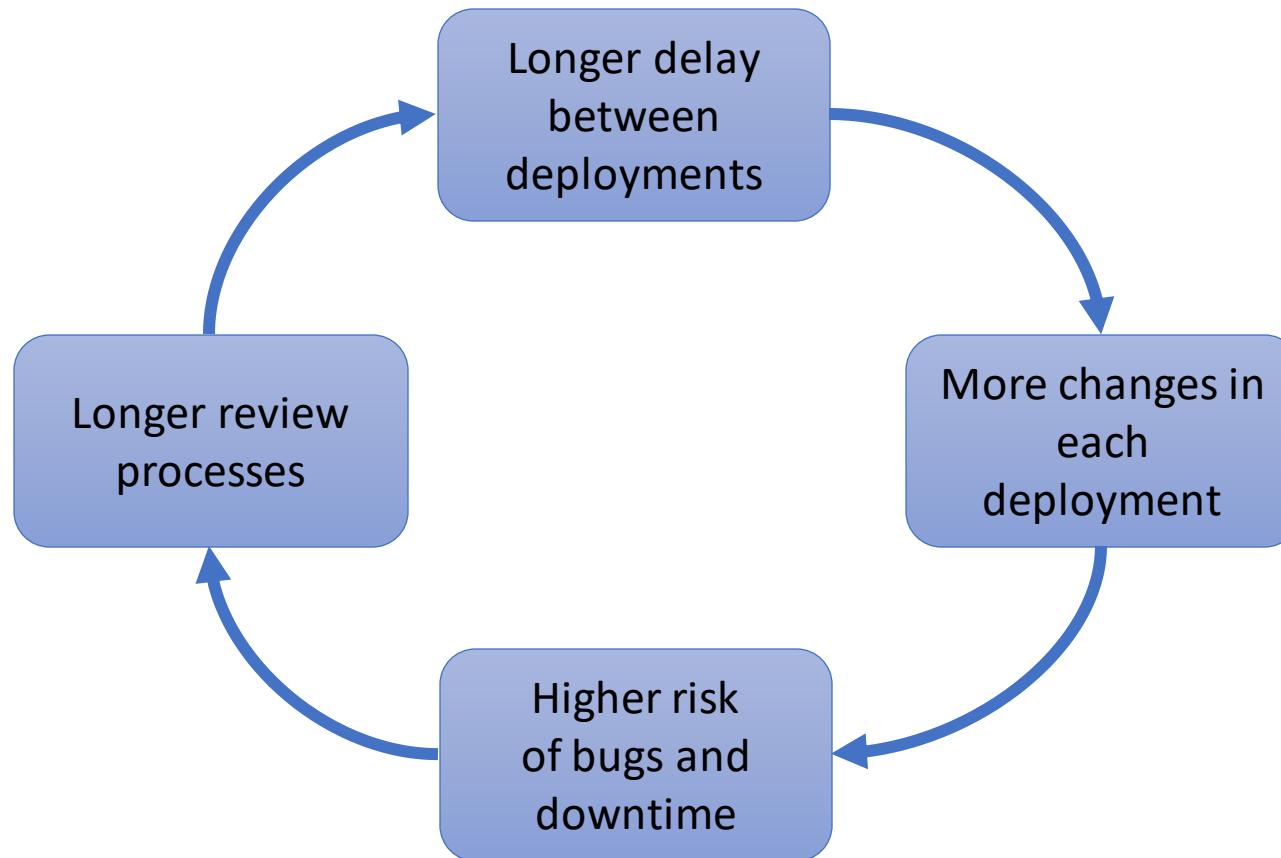
Facilitates re-configuration and error checking

## Automatic and continuous

Goal: Increase teams agility

# Be careful with manual deployments

Vicious circle of deployment size and risk



# Continuous deployment

- *"If it hurts do it more often"*
- In the limit: "Do everything continuously"
- Run the full pipeline in every commit
- Final stage: deployment in production
  - Possibilities
    - Confirmation by some human before going to production
    - Automatic deployment to production
    - Deployment to production marked by some tags
  - Trade-off
    - Cost of moving slower vs cost of error in deployment

# Virtual machines

- Running apps on virtual machines
  - Require operating system + libraries
  - Isolate apps from specific hardware
  - Cloud virtual machine providers: Azure, AWS, Google, Alibaba, ...
- Azure example
  - <https://portal.azure.com/#home>

# What is Docker?

Platform for developers and system administrators

Started in 2011

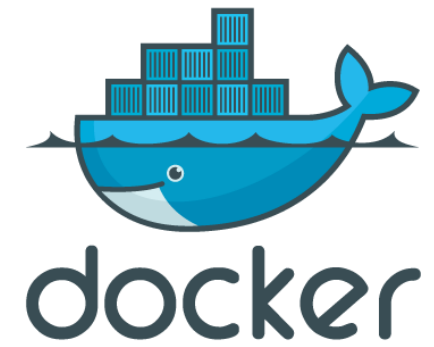
Based on containers and images

## Several parts

- Specification for container descriptions (images)

- Platform that runs containers

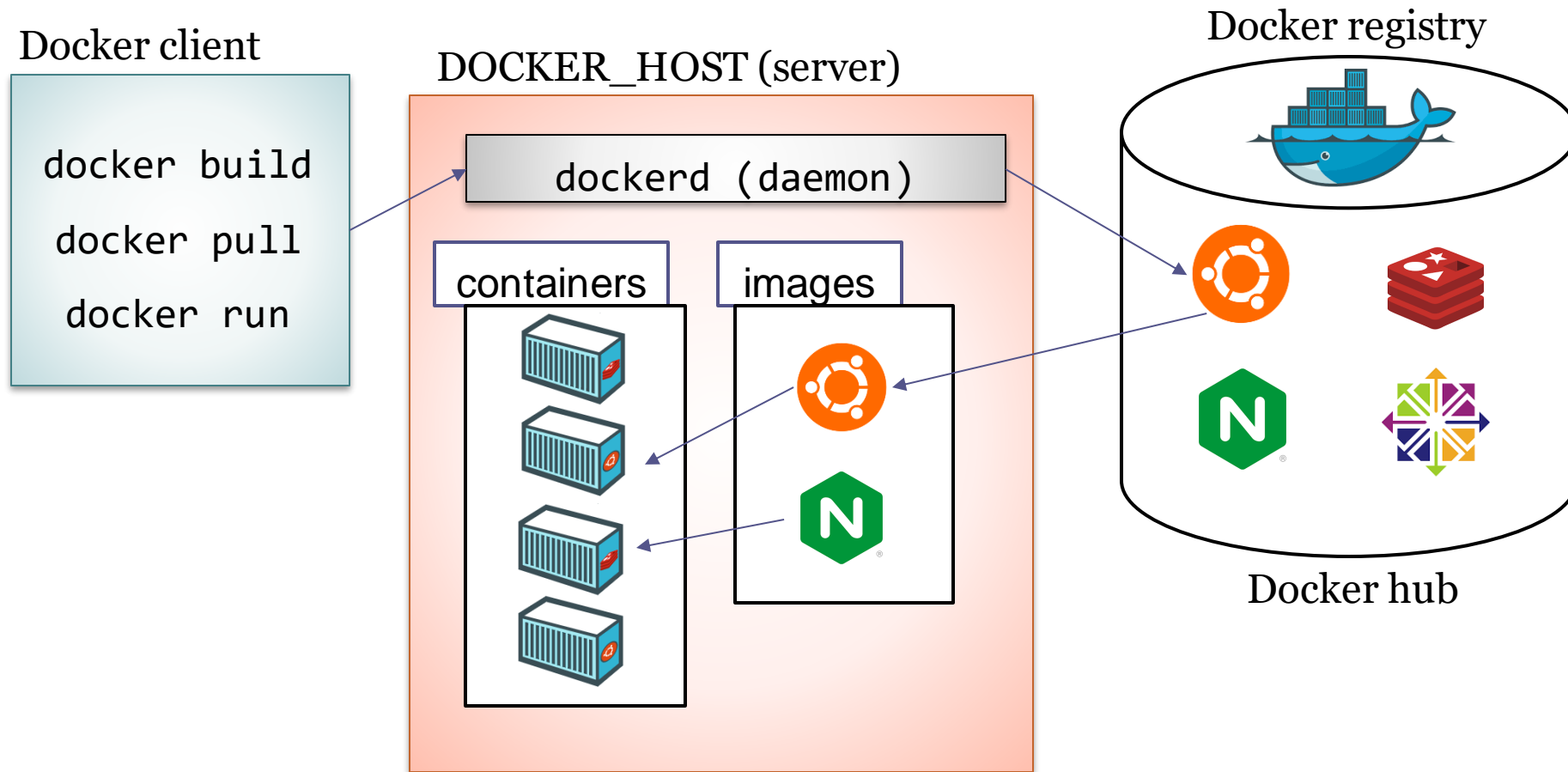
- Container registry (Docker-hub)





# Docker high-level architecture

## Client-server architecture



# What is an image?

A file that can be used to create a runnable package

Includes all things necessary to run the application:

Code

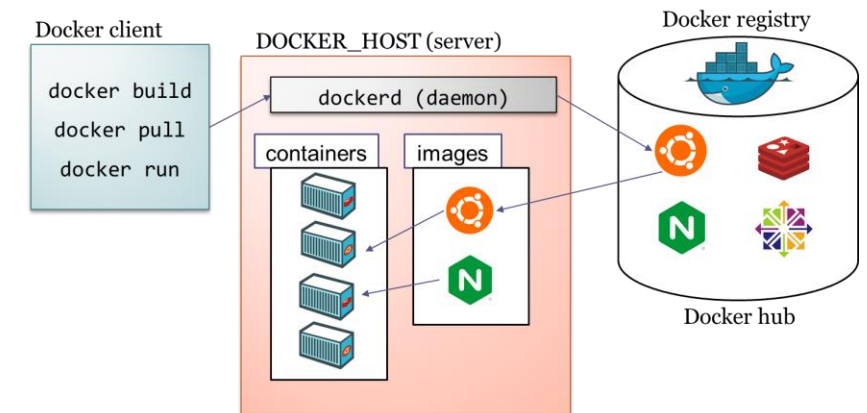
Runtime system

Libraries

Runtime variables

Configuration files

It doesn't have state and never changes



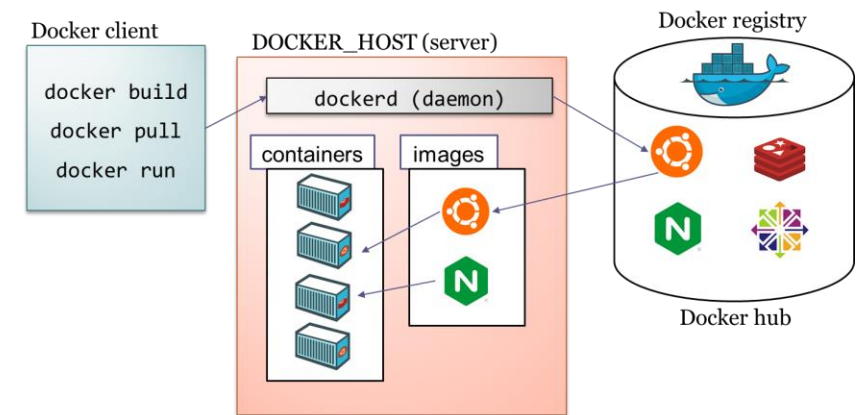
# What is a container?

It is a live instance of an image

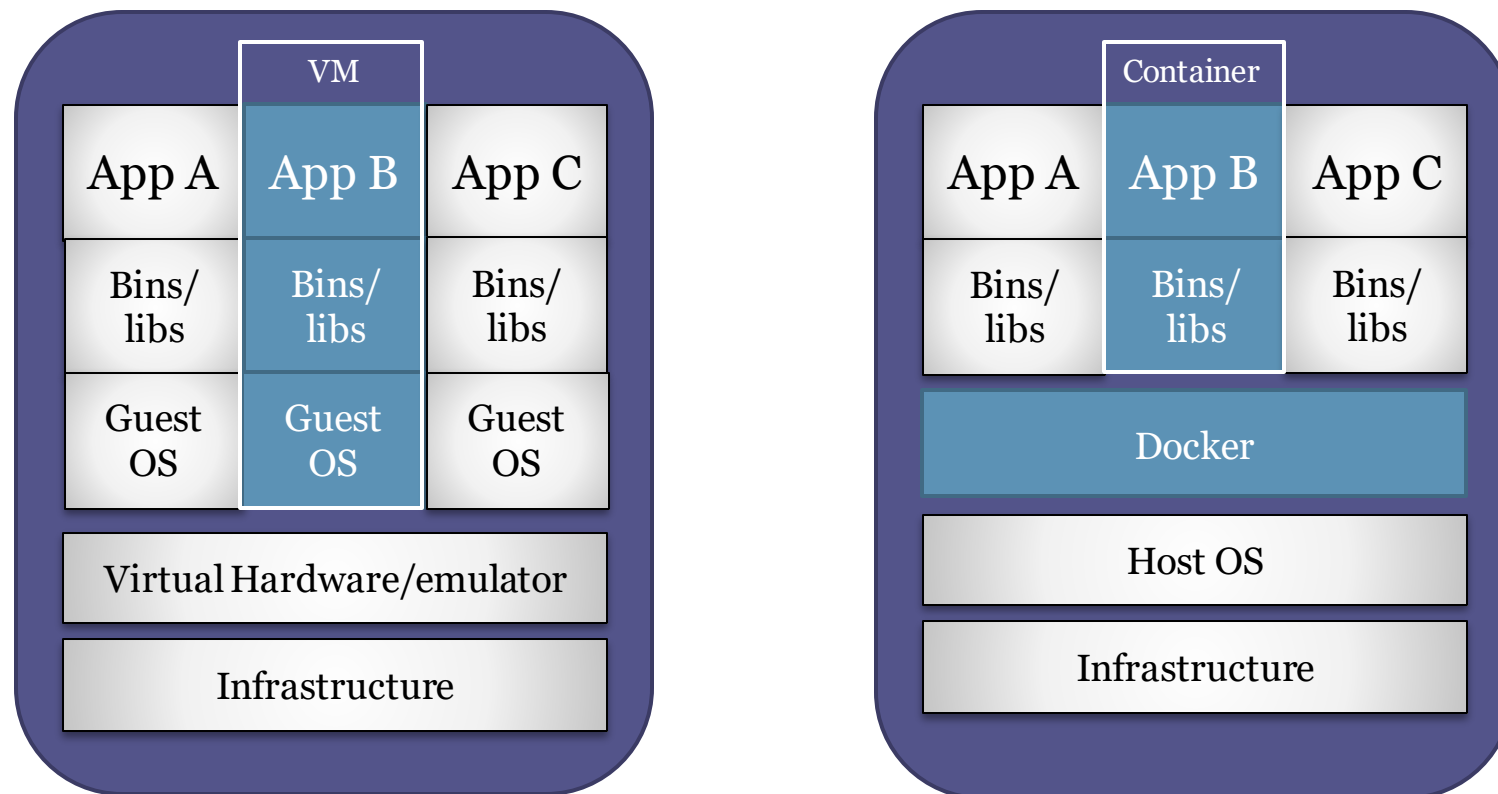
Docker is based on containers that enclose applications

Docker allows orchestration between containers

Linking several containers we can make a complex architecture



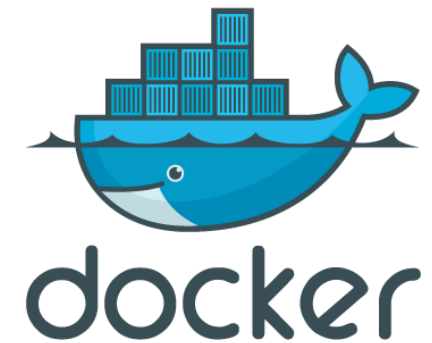
# Containers vs Virtual machines



Source: <https://docs.docker.com/get-started/#containers-and-virtual-machines>  
<https://stackoverflow.com/questions/16047306/how-is-docker-different-from-a-virtual-machine>

# Obtaining docker

- <https://www.docker.com>
- Available for GNU/Linux, windows and Mac
- Docker desktop (Windows/Mac)
- Docker ToolBox [faq#issue3](#)



# Docker image registries

## Docker Hub

Docker image repository <https://hub.docker.com/>

Example: Need a web-server for development

```
docker pull nginx
```

```
docker pull httpd
```

## Github Container Registry (<https://ghcr.io/>)

Previously called github packages

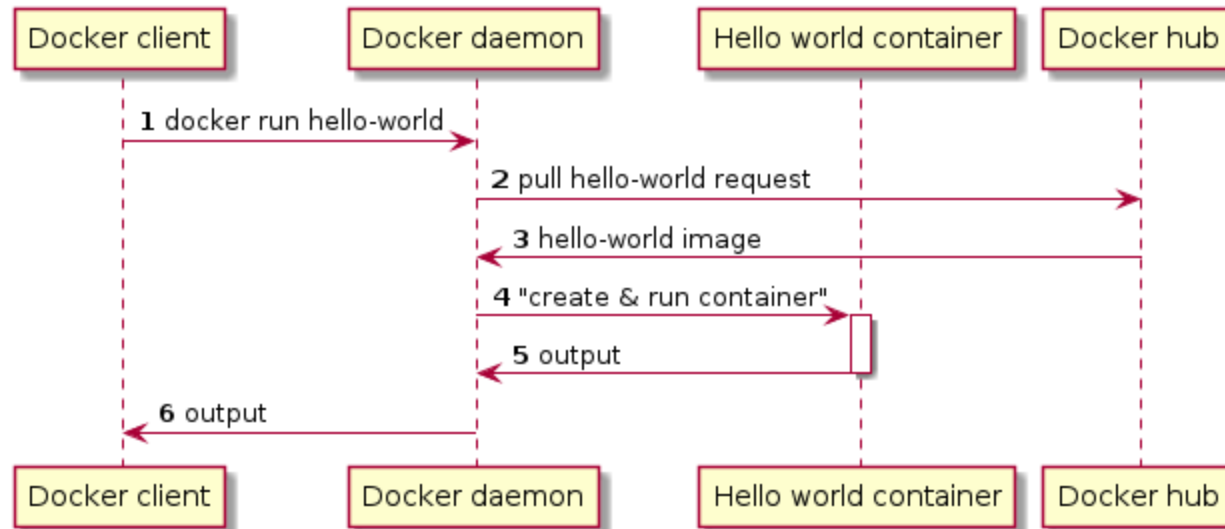
# Docker step by step

## Install Docker

```
$ docker -v
```

## Run "Hello World"

```
$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
1b930d010525: Pull complete
Digest: sha256:f9dfddf63636d84ef479d645ab5885156ae030f...
Status: Downloaded newer image for hello-world:latest
```



# Docker example running Linux

## Run Ubuntu

```
$ docker run -it ubuntu:latest /bin/bash
. . .
root@813cb77cebb2:/# ls -la
total 72
drwxr-xr-x    1 root root 4096 Mar 30 05:46 .
drwxr-xr-x    1 root root 4096 Mar 30 05:46 ..
-rwxr-xr-x    1 root root    0 Mar 30 05:46 .dockerenv
drwxr-xr-x    2 root root 4096 Mar 11 21:05 bin
drwxr-xr-x    2 root root 4096 Apr 24  2018 boot
drwxr-xr-x    5 root root  360 Mar 30 05:47 dev
drwxr-xr-x    1 root root 4096 Mar 30 05:46 etc
. . .
drwxr-xr-x    1 root root 4096 Mar 11 21:03 usr
drwxr-xr-x    1 root root 4096 Mar 11 21:05 var
root@813cb77cebb2:/#
```



# Docker status

## Commands to check status

```
λ docker image ls
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hello-world   latest    fce289e99eb9   14 months ago  1.84kB

λ docker container ls --all
CONTAINER ID   IMAGE          COMMAND         CREATED        STATUS
8b6518da11db   hello-world    "/hello"        9 minutes ago  Exited (0) 9 minutes ago
```

[https://github.com/pglez82/docker\\_cheatsheet](https://github.com/pglez82/docker_cheatsheet)

# Docker simple web server

## Run a web server with Docker

Run in background

publish:expose port

```
$ docker run --detach --publish=80:80 --name=webserver nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
68ced04f60ab: Pull complete
28252775b295: Pull complete
a616aa3b0bf2: Pull complete
Digest: sha256:2539d4344dd18e1df02be842ffc435f8e1f699cfc55516e2cf2cb16b7a9aea0b
Status: Downloaded newer image for nginx:latest
b7e9213eb3367cd465b29701a7e6441a7210e46d42010cd20e76ddde0e72ee280
```

localhost

## Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

# Some commands

```
docker info
```

```
docker ps
```

```
docker image ls
```

```
docker container ls -all
```

```
docker pull
```

```
docker run
```

```
docker stop
```

```
docker rm
```

# How to build an image

DSL to build images

We need to create a file, called **Dockerfile**

It contains commands necessary to build the image

**Keywords:** FROM, RUN, ADD, COPY, ENV, EXPOSE, CMD...

Dockerfile

```
FROM ubuntu
```

```
CMD echo "Hi Software architecture students"
```

# Building an image

1. Create a folder for the project
2. Edit a Dockerfile (no extension)
3. `docker build -t image_name .`
4. `docker images` (list images)
5. `docker run image_name`

## Dockerfile

```
FROM ubuntu
CMD echo "Hi ASW students"
```

```
λ docker build -t "example1" .
Sending build context to Docker daemon 2.048kB
Step 1/2 : FROM ubuntu
latest: Pulling from library/ubuntu
5bed26d33875: Pull complete
...
Digest: sha256:bec5a2727be7fff3d308193cfde3491f8fba1a2...
Status: Downloaded newer image for ubuntu:latest
--> 4e5021d210f6
Step 2/2 : CMD echo "Hi Software architecture students"
--> Running in 9d5516995c2b
Removing intermediate container 9d5516995c2b
--> 41784c740df4
Successfully built 41784c740df4
Successfully tagged example1:latest
```

```
λ docker images
REPOSITORY TAG      IMAGE ID  CREATED          SIZE
example1   latest  41784c740 32 seconds ago  64.2MB
```

```
λ docker run example1
Hi ASW students
```

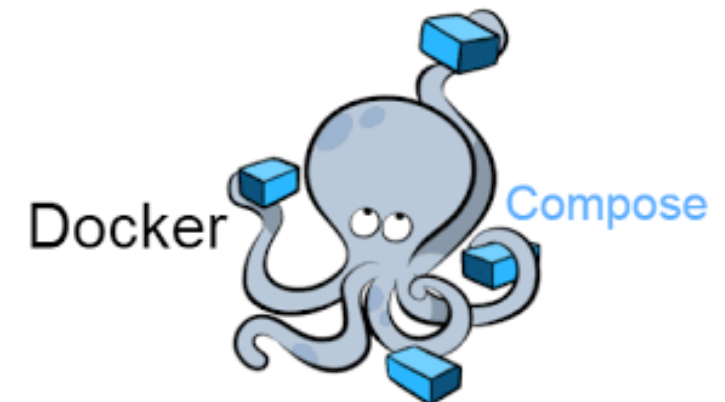
# Combining multiple docker containers

Docker compose allows modularization of an application or architecture

Different services are defined that communicate among them

Each service is in a separate container

File: docker-compose.yml



# Running Docker compose

## Configuration

- We can configure multiple services
- Each service can depend on others
- By default, all services share the same network and are accessible through their container name

## Running

- For running (or stopping) a docker-compose file we execute:  
`docker-compose (up|down)`

# Github actions

It allows to run automatically *workflows*

From some actions

Like for each commit, each release,...

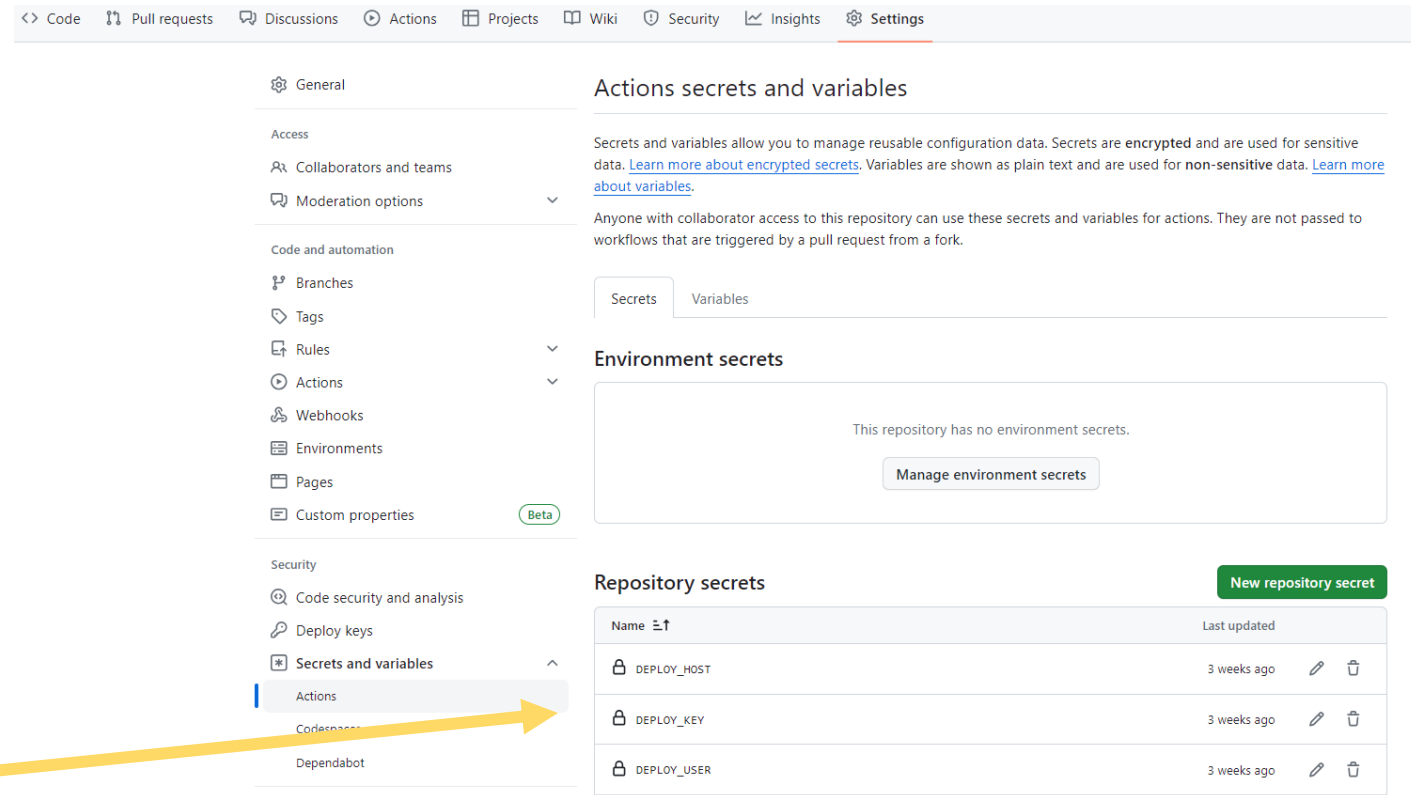
Configuration: YAML files in `.github/workflows`

Example: [https://github.com/Arquisoft/wiq\\_0/tree/master/.github/workflows](https://github.com/Arquisoft/wiq_0/tree/master/.github/workflows)



# Runtime environment variables

- They allow to parameterize the deployment
  - Defining credentials, identifiers,...
- Github allows to define SECRETS in each repository



The screenshot shows the GitHub repository settings page. The left sidebar has a menu with items like General, Access, Code and automation, Security, and Secrets and variables. The 'Secrets and variables' item is highlighted. The main content area is titled 'Actions secrets and variables' and contains sections for 'Environment secrets' (which is empty) and 'Repository secrets' (which contains a table of secrets).

Name	Last updated
DEPLOY_HOST	3 weeks ago
DEPLOY_KEY	3 weeks ago
DEPLOY_USER	3 weeks ago

In wiq\_XXX:

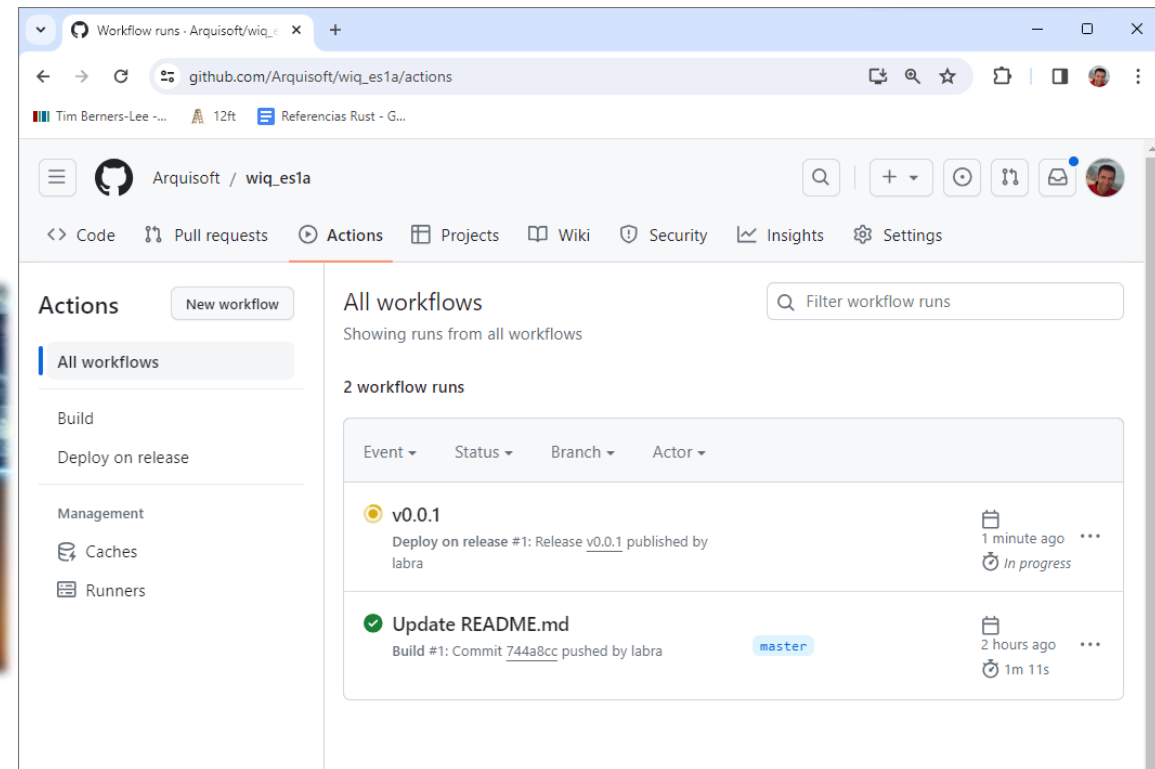
DEPLOY\_HOST: IP of virtual machine

DEPLOY\_USER: User that can have ssh Access to the virtual machine

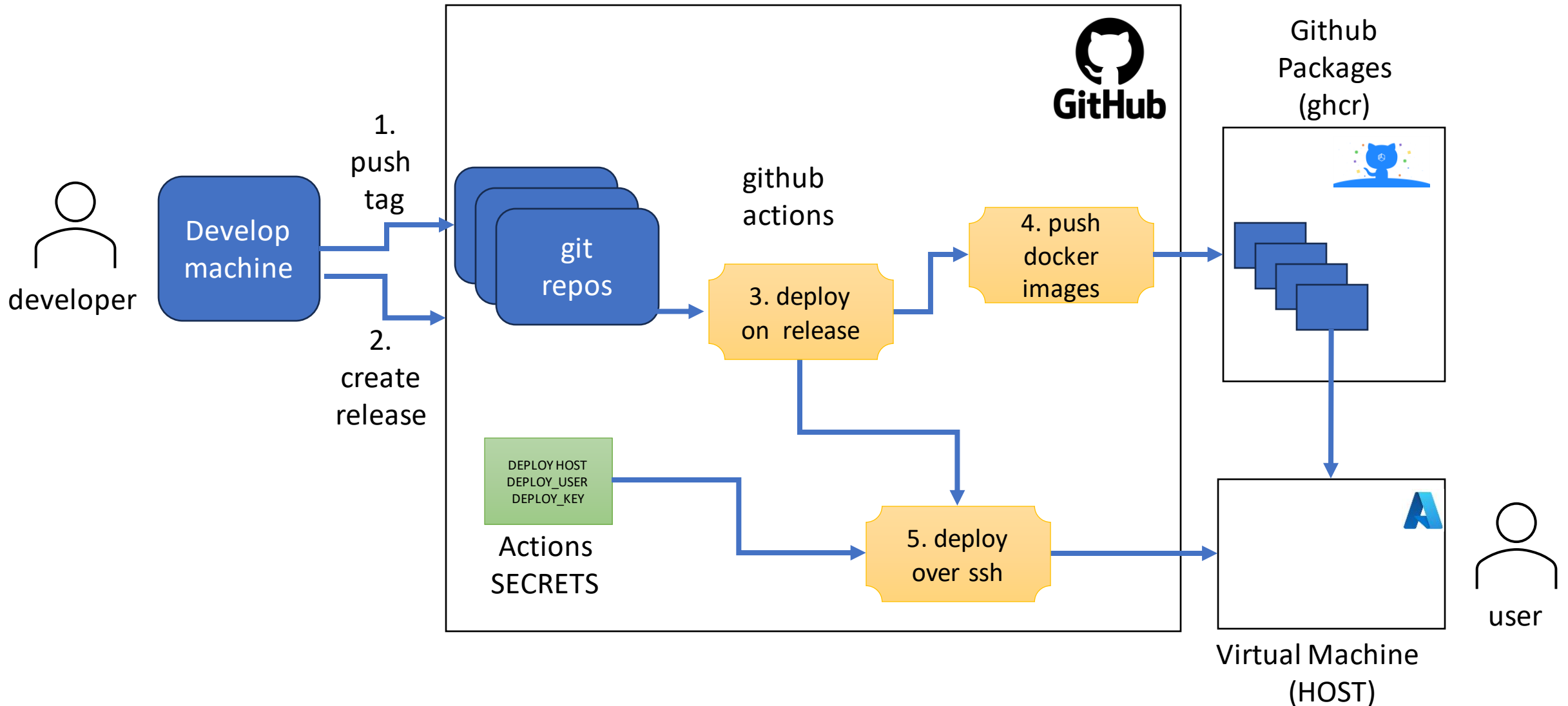
DEPLOY\_KEY: Private key of that user

# Continuous deployment for each release

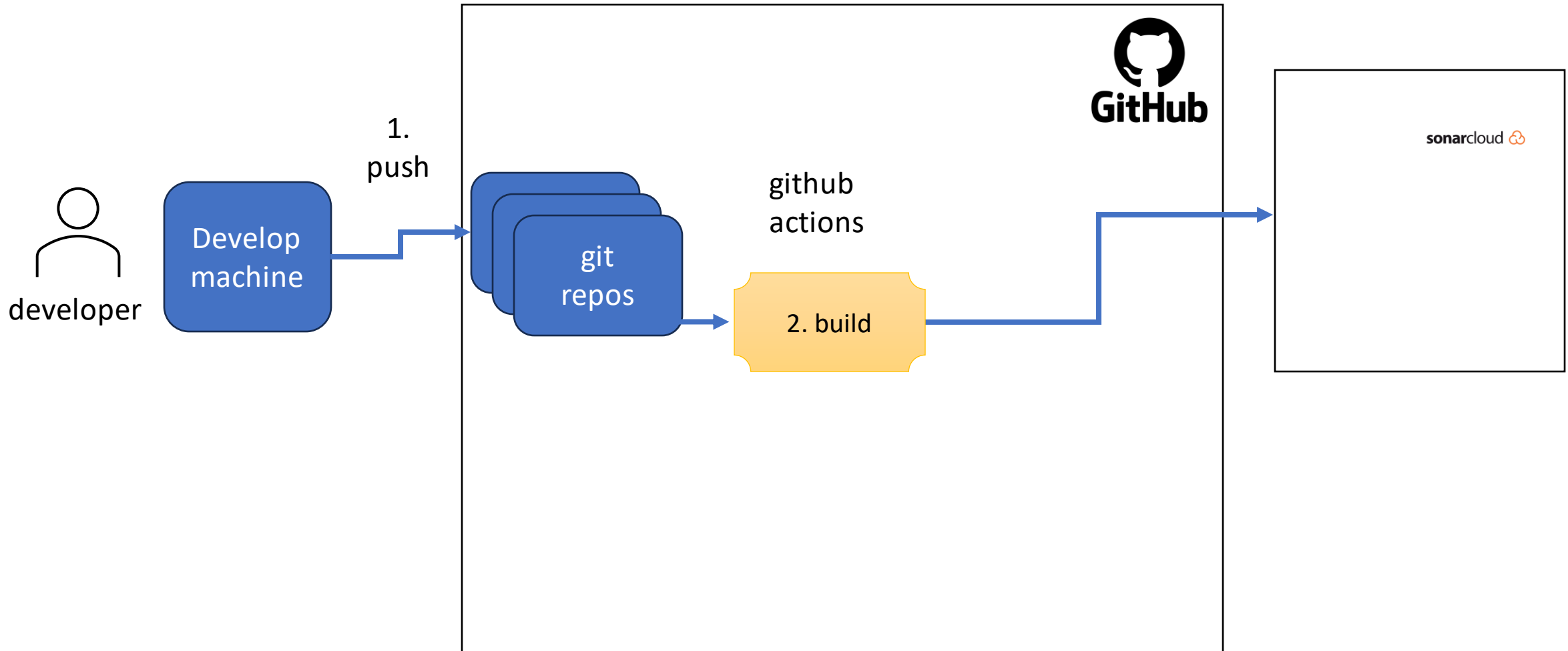
- Steps
  - Create a tag and push to github
    - `git tag -a v0.0.1 -m "v0.0.1"`
    - `git push origin v0.0.1`
  - Create release from github
- Watch automatic deployment



# Deployment pipeline in wiq\_xxx



# Deployment pipeline in wiq\_xxx



# Extra information

Small repository with all the basic commands used in docker:

[https://github.com/pglez82/docker\\_cheatsheet](https://github.com/pglez82/docker_cheatsheet)