

# Environment manager from your OS to your environment

## Encapsulation levels using docker, conda



P. Marin, M. Hiriart, P. Ruiz & N. Goué  
[aubi@uca.fr](mailto:aubi@uca.fr)

Université Clermont Auvergne, AuBi, Mésocentre

18 novembre 2022



UNIVERSITÉ  
Clermont Auvergne

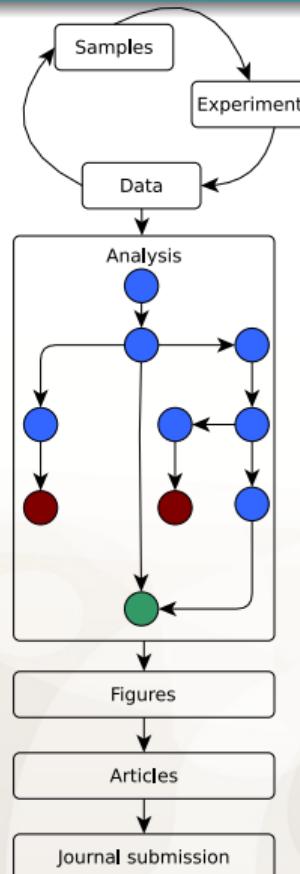


This work is derived from the IFB and I2BC team members

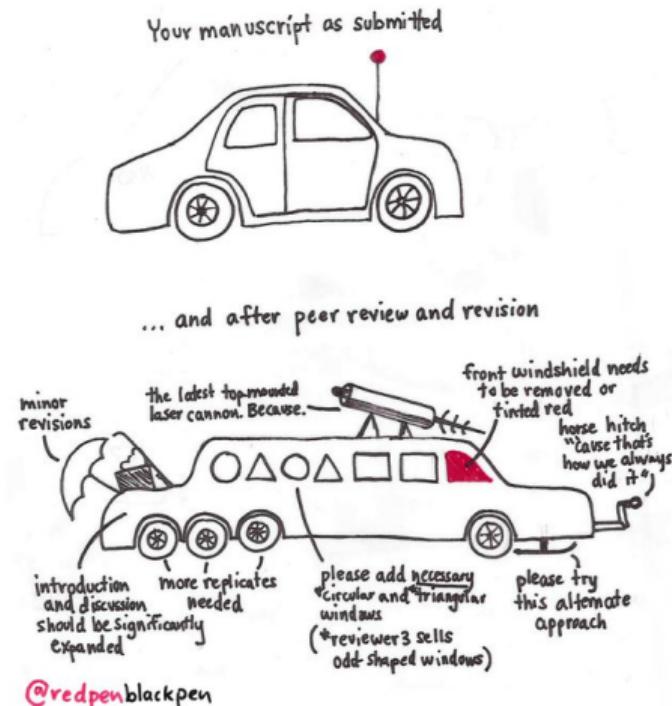
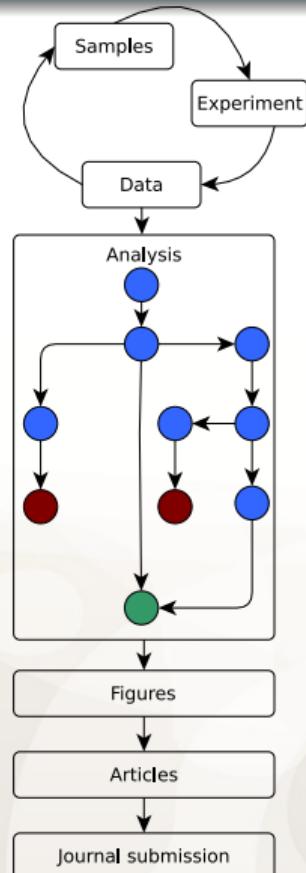
# Sommaire

- 1 A common use-case
  - Retry my results
  - The use of packaging
  - Example with R
- 2 Manage your local environment
  - How conda works
- 3 Manage your hardware configuration
- 4 How virtual manager works
- 5 Manage your OS configuration
  - How container works
- 6 Conda ecosystem
  - a case of bioconda
- 7 A second level, the containers
  - Docker
  - Singularity

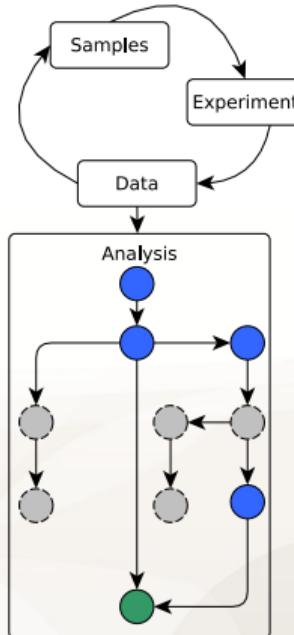
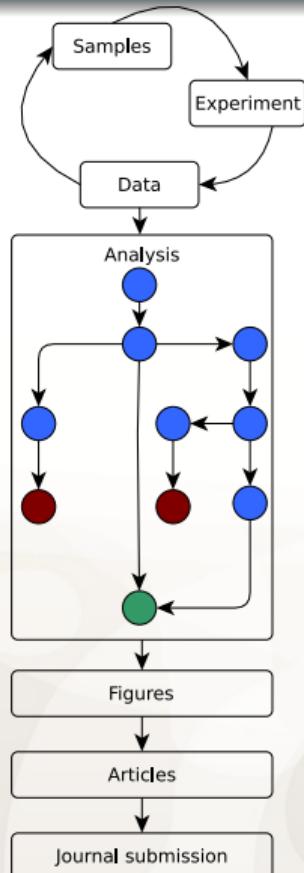
# A classic use case



# A classic use case



# A classic use case



# A classic use case

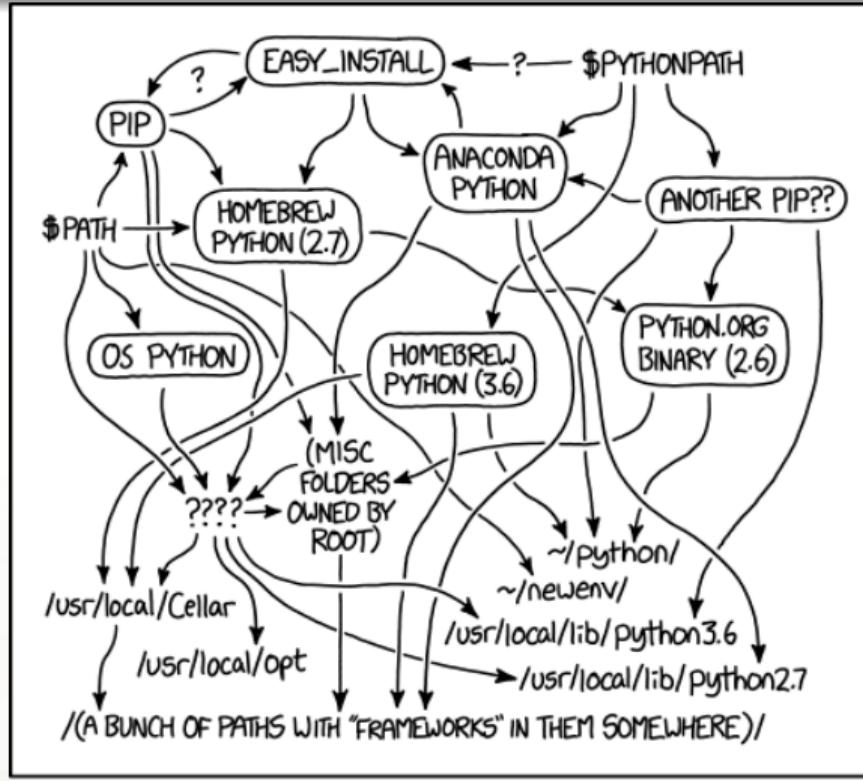
What are the changes ?

- Tool version
- Packages
- Environment variables
- OS version
- The computer
- ...

## ■ Tool compatibility troubles

- Python version ? 2.7, 3.8...
- Which tool version ?
- Installation without root access
- coexistence between several versions, libraries

# My python env



MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED  
THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

# Encapsulation levels

*Encapsulation : capture the environment of applications (OS, packages, libraries) to control their execution*



- Hardware virtualisation (virtual machines)

# Encapsulation levels

*Encapsulation : capture the environment of applications (OS, packages, libraries) to control their execution*



- Hardware virtualisation (virtual machines)



- OS virtualisation (images and containers)

# Encapsulation levels

*Encapsulation : capture the environment of applications (OS, packages, libraries) to control their execution*



- Hardware virtualisation (virtual machines)



- OS virtualisation (images and containers)
- Environment management (package manager) **CONDA**

# Example of R and package installation

## Classical installation

- Start with a computer and a specific OS

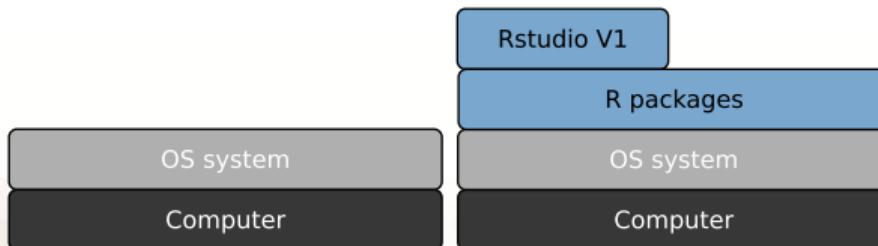
OS system

Computer

# Example of R and package installation

## Classical installation

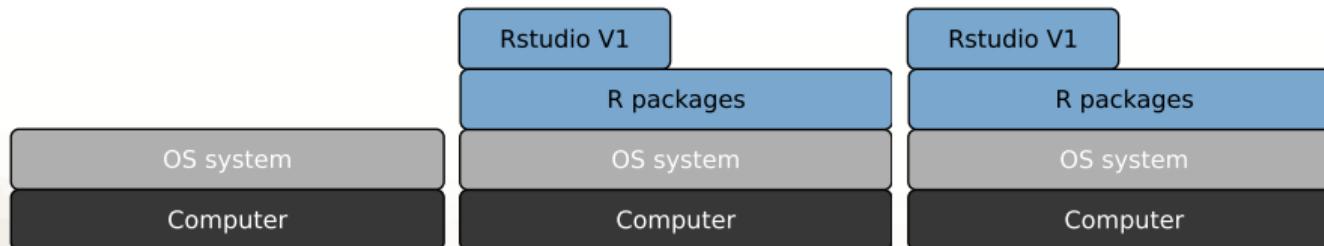
- Start with a computer and a specific OS
- Inside, we installed a new  application



# Example of R and package installation

## Classical installation

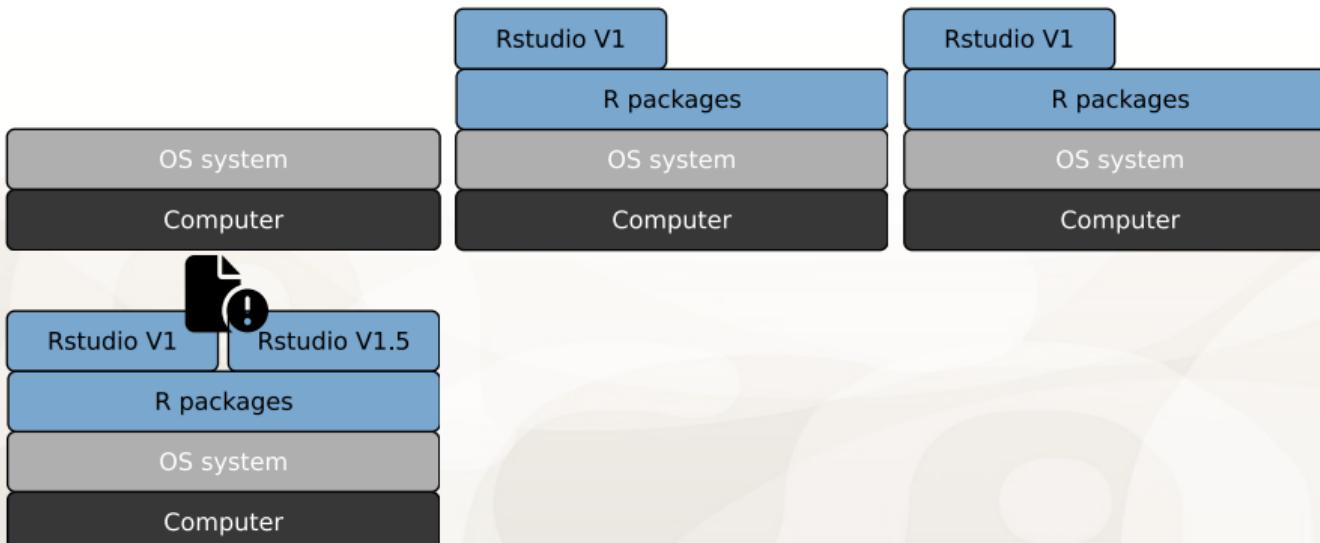
- Start with a computer and a specific OS
- Inside, we installed a new  application
-  need some dependencies



# Example of R and package installation

## Classical installation

- Start with a computer and a specific OS
- Inside, we installed a new  application
-  need some dependencies
- we tested the last  version -> might be conflicts



# Example of R and package installation

## Conda use

- The idea is to separate each application in here own environment **CONDA**

Conda env

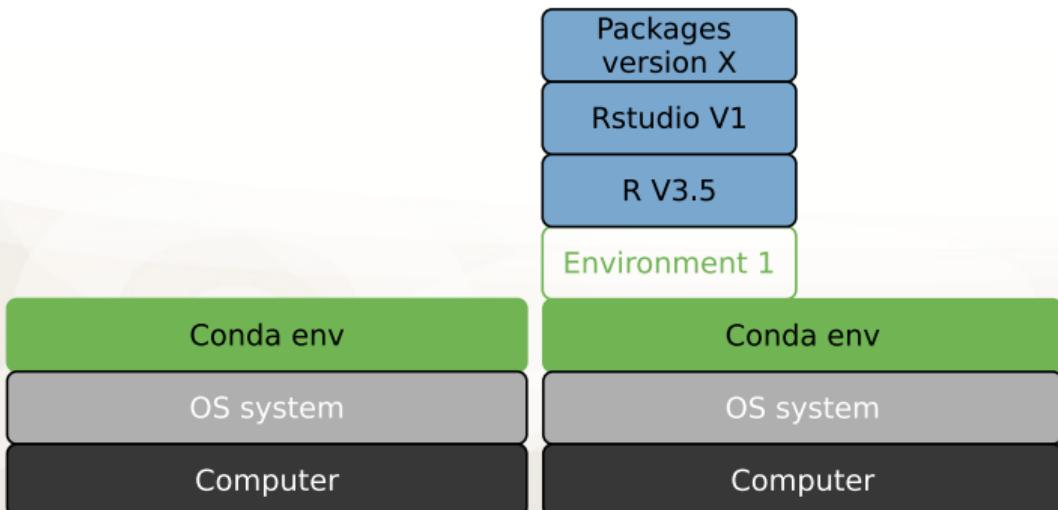
OS system

Computer

# Example of R and package installation

## Conda use

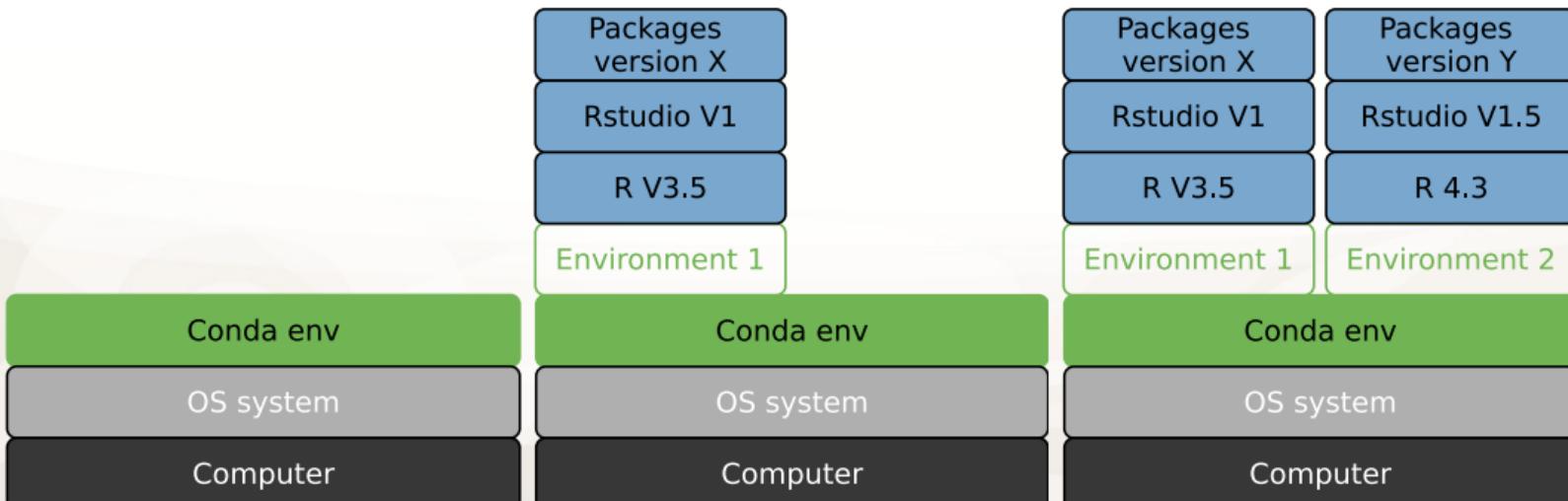
- The idea is to separate each application in here own environment **CONDA**
- A tool version, a conda environment



# Example of R and package installation

## Conda use

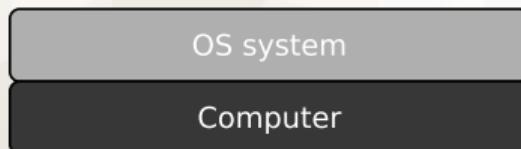
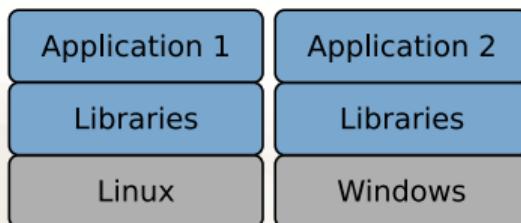
- The idea is to separate each application in here own environment **CONDA**
- A tool version, a conda environment
- Create a new environment for my new tool version, my analysis...



# Example of R and package installation

## hardware virtualisation

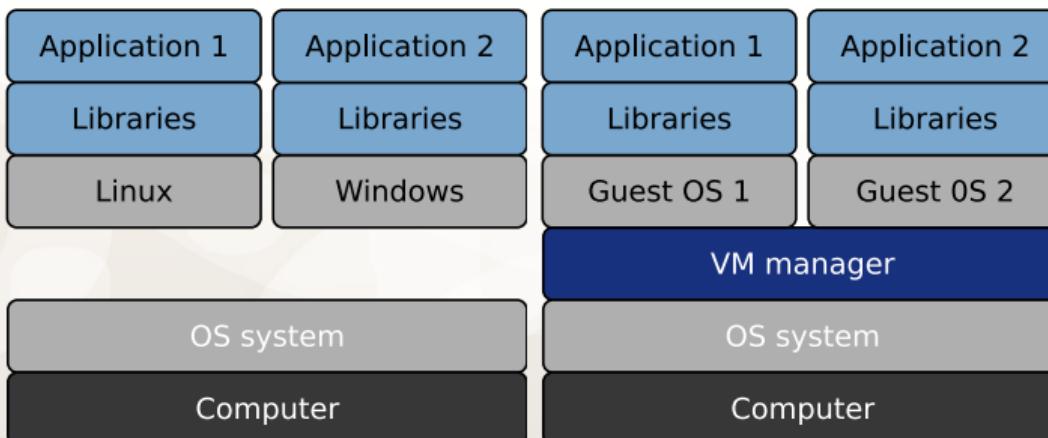
- If we want a software from a different OS ?



# Example of R and package installation

## hardware virtualisation

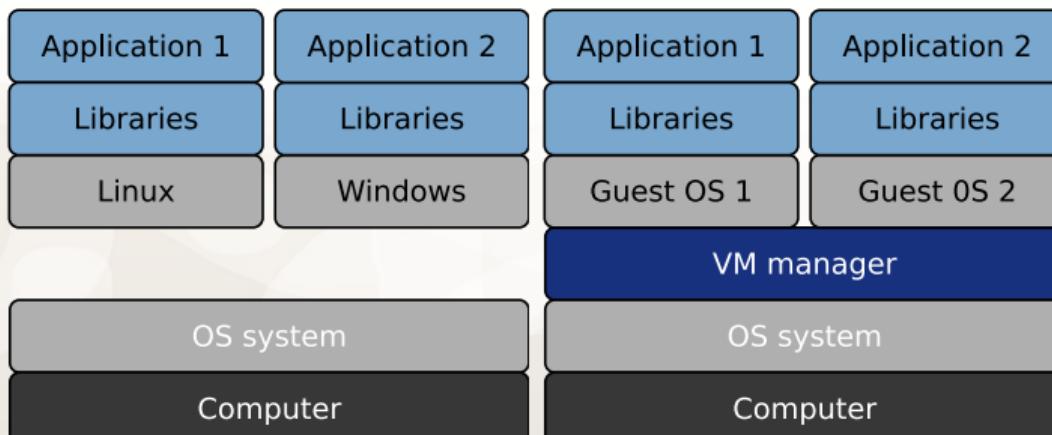
- If we want a software from a different OS ?
- Use virtual machines



# Example of R and package installation

## hardware virtualisation

- If we want a software from a different OS ?
- Use virtual machines
- Each application get a total different and independant environment



# Example of R and package installation

## hardware virtualisation

- If we want a software from a different OS ?
- Use virtual machines
- Each application get a total different and independant environment
- Virtual machine could be transferred to another computer

# Example of R and package installation

## hardware virtualisation

- If we want a software from a different OS ?
- Use virtual machines
- Each application get a total different and independant environment
- Virtual machine could be transferred to another computer
- Redundancy between VMs

# Example of R and package installation

## hardware virtualisation

- If we want a software from a different OS ?
- Use virtual machines
- Each application get a total different and independant environment
- Virtual machine could be transferred to another computer
- Redundancy between VMs
- Heavy to set up

# Example of R and package installation

## hardware virtualisation

- If we want a software from a different OS ?
- Use virtual machines
- Each application get a total different and independant environment
- Virtual machine could be transferred to another computer
- Redundancy between VMs
- Heavy to set up
- No automation

Oracle VM VirtualBox Manager

Tools      New      Settings      Discard      Show ▾

Icon	Name	Status
	ovs34-efi	Powered Off
	ubuntu-18.04	Running
	win2016srv	Running
	centos7	Powered Off
	OracleLinux7	Running
	OracleLinux6	Powered Off
	ol7-vbox6	Powered Off

**General**

Name: ubuntu-18.04  
Operating System: Ubuntu (64-bit)  
Settings File Location: /Users/scr/VirtualBox/ubuntu-18.04

**System**

Base Memory: 4096 MB  
Boot Order: Optical, Hard Disk  
Acceleration: VT-x/AMD-V, Nested Paging, KVM Paravirtualization

**Preview**

**General**

Name: win2016srv  
Operating System: Windows 2016 (64-bit)  
Settings File Location: /Users/scr/VirtualBox/win2016srv

**System**

Base Memory: 8192 MB  
Boot Order: Hard Disk, Optical, Floppy  
Acceleration: VT-x/AMD-V, Nested Paging, PAE/NX

**Preview**

**General**

Name: OracleLinux7  
Operating System: Oracle (64-bit)  
Settings File Location: /Users/scr/VirtualBox/OracleLinux7

**System**

Base Memory: 8192 MB  
Processors: 2  
Boot Order: Hard Disk, Optical  
Acceleration: VT-x/AMD-V, Nested Paging, PAE/NX, KVM Paravirtualization

**Preview**

RSITÉ  
mont Auvergne

# Example of R and package installation

## OS virtualisation

- "Trick" applications into believing that they are in a different OS than the host's

# Example of R and package installation

## OS virtualisation

- "Trick" applications into believing that they are in a different OS than the host's
- Named containers :  docker or 

# Example of R and package installation

## OS virtualisation

- "Trick" applications into believing that they are in a different OS than the host's
- Named containers :  docker or 
- Avoid redundancy

OS system

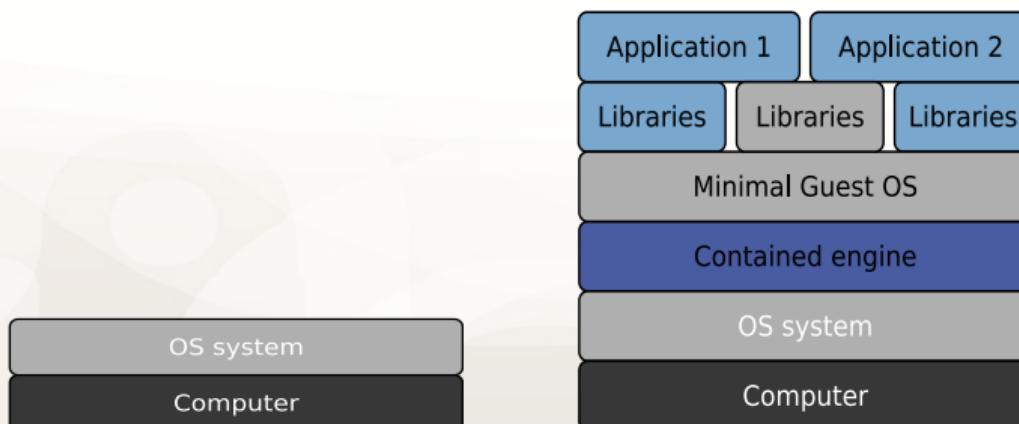
Computer

# Example of R and package installation

## OS virtualisation

- "Trick" applications into believing that they are in a different OS than the host's

- Named containers :  or 
- Avoid redundancy



# Example of R and package installation

## OS virtualisation

- "Trick" applications into believing that they are in a different OS than the host's

- Named containers :



- Avoid redundancy

- Speed
  - Faster installation
  - No boot time

# Example of R and package installation

## OS virtualisation

- "Trick" applications into believing that they are in a different OS than the host's

■ Named containers :  docker or 

- Avoid redundancy
- Speed
  - Faster installation
  - No boot time
- Lightweight
  - Minimal base OS
  - Minimal set of library and global environment
  - Easy sharing of application



rstudio

Explore Pricing Sign In

Register

## Filters

## Products

 Images Extensions Plugins

## Trusted Content

 Docker Official Image ⓘ Verified Publisher ⓘ Sponsored OSS ⓘ

## Operating Systems

 Linux Windows

## Architectures

 ARM ARM 64 IBM POWER IBM Z PowerPC 64 LE

1 - 25 of 2 061 results for rstudio.

Best Match



ibmcom/rstudio-ppc64le

VERIFIED PUBLISHER

By IBM • Updated 3 years ago

Integrated development environment (IDE) for R

Linux ppc64le

487 4

Downloads Stars



wholelake/rstudio-base

SPONSORED OSS

By wholelake • Updated 4 years ago

Linux x86-64

127 0

Downloads Stars



bioconductor/rstudio\_yescd8

SPONSORED OSS

By bioconductor • Updated 16 days ago

Linux x86-64

23 0

Downloads Stars



truecharts/rstudio

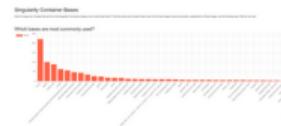
SPONSORED OSS

By truecharts • Updated 11 days ago

Linux x86-64

5 0

Downloads Stars



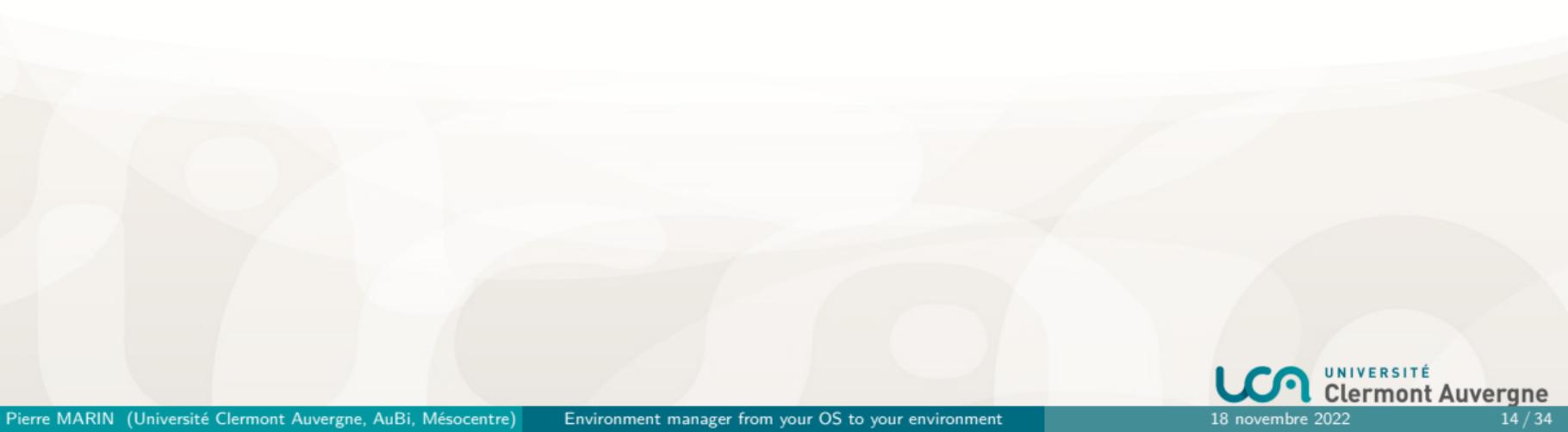
All repos 1334

rstudio

Stack	Description	Singularity* recipe	Topics	QC	Stars	Watchers
grst/rstudio-server-conda	Run Rstudio Server in a conda environment	0		license MIT last commit October 2021	106	3
nickjer/singularity-rstudio	RStudio Server in a Singularity container	0.1	rstudio-server, singularity-image	license MIT last commit June 2021	40	5
singularityhub/singularity-compose-examples	A simple example of running a MongoDB instance to query a database	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	singularity-compose, mongodb	license not specified last commit August	12	5
OSC/bc_osc_rstudio_server	Batch Connect - OSC RStudio Server	0		license MIT last commit June	6	10
RBigData/singularity	Singularity configurations for R and pbDR packages.	0 1 2 3 4 5 6 7 8		license BSD-2-Clause last commit October 2019	3	4

- Docker is not easy to use on a cluster system
- Docker private company policies

# Conda system



# Conda system

## ■ Anaconda

- Open source distribution
- Cross platform
- Available on cluster without admin whrite
- Thousands of available tool in informatic and bioinformatic

# Conda system

## ■ Anaconda

- Open source distribution
- Cross platform
- Available on cluster without admin whrite
- Thousands of available tool in informatic and bioinformatic

## ■ Miniconda

- A lightheight Anaconda version with minimal requirment
- Same advantages ad Anaconda

# Conda system

## ■ Anaconda

- Open source distribution
- Cross platform
- Available on cluster without admin whrite
- Thousands of available tool in informatic and bioinformatic

## ■ Miniconda

- A lightheight Anaconda version with minimal requirment
- Same advantages ad Anaconda

## ■ Conda **CONDA**

- Package manager AND environment manager
- installed with Ana or Miniconda
- Python based but can also install tools from R, C++ or Julia...

# Conda system

CONDA CHEATSHEET	
QUICK START	
Tip: It is recommended to create a new environment for any new project or workflow.	
verify conda install and check version	conda info
update conda in base environment	conda update -n base conda
install latest anaconda distribution (see <a href="#">release notes</a> )	conda install anaconda=2022.08
create a new environment (tip: name environment descriptively)	conda create --name ENVNAME
activate environment (do this before installing packages)	conda activate ENVNAME
CHANNELS AND PACKAGES	
Tip: Package dependencies and platform specifics are automatically resolved when using conda.	
install packages from specified channel	conda install -c CHANNELNAME PKG1 PKG2
list installed packages	conda list
uninstall package	conda uninstall PKGNAME
update all packages	conda update --all
install specific version of package	conda install PKGNAME=3.1.4
install a package from specific channel	conda install CHANNELNAME::PKGNAME
install package with AND logic	conda install "PKGNAME>2.5,<3.2"
install package with OR logic	conda install "PKGNAME [version='2.5 3.2']"
list installed packages with source info	conda list --show-channel-urls
view channel sources	conda config --show-sources
add channel	conda config --add channels CHANNELNAME
set default channel for pkg fetching (targets first channel in channel sources)	conda config --set channel_priority strict
WORKING WITH CONDA ENVIRONMENTS	
Tip: List environments at the beginning of your session. Environments with an asterisk are active.	

# The channels and the tools

The tools are packaged and available on several **channels**

. Bioconda : sustainable and comprehensive software distribution for the life sciences *Grüning et al.*, Nature methods, 2018. DOI 10.1038/s41592-018-0046-7

# The channels and the tools

The tools are packaged and available on several **channels**

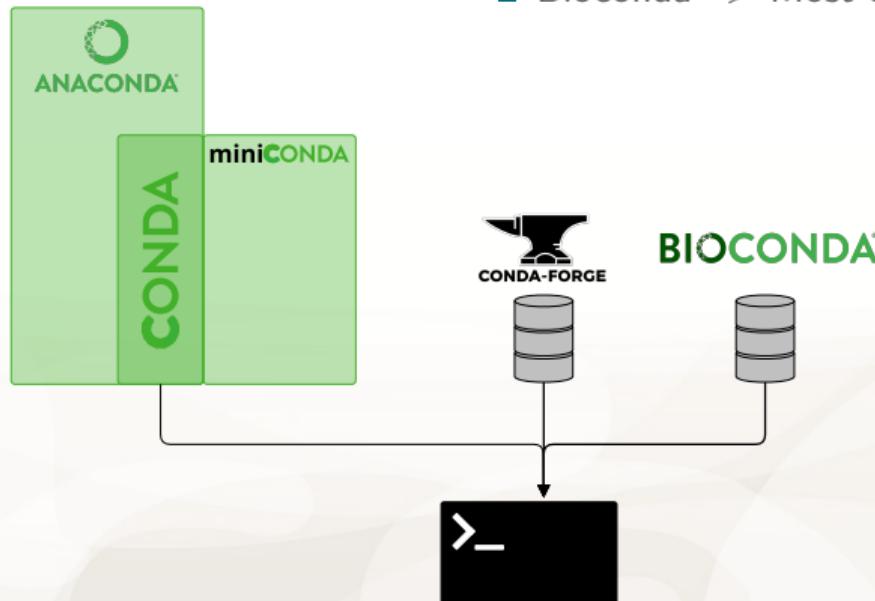
- Conda-forge
- Anaconda
- R
- Bioconda -> Most of the bioinformatic tools

. Bioconda : sustainable and comprehensive software distribution for the life sciences Grüning et al., Nature methods, 2018. DOI 10.1038/s41592-018-0046-7

# The channels and the tools

The tools are packaged and available on several **channels**

- Conda-forge
- Anaconda
- R
- Bioconda -> Most of the bioinformatic tools



. Bioconda : sustainable and comprehensive software distribution for the life sciences Grüning et al., Nature methods, 2018. DOI 10.1038/s41592-018-0046-7

# Basic commands

## Create environment

```
$ conda create -n [env_name]  
$ conda activate [env_name]  
$ conda deactivate [env_name]
```

# Basic commands

## Create environment

```
$ conda create -n [env_name]  
$ conda activate [env_name]  
$ conda deactivate [env_name]
```

## list environment or packages

```
$ conda env list  
$ conda list
```

# Basic commands

## Create environment

```
$ conda create -n [env_name]  
$ conda activate [env_name]  
$ conda deactivate [env_name]
```

## list environment or packages

```
$ conda env list  
$ conda list
```

## Search tools

```
$ conda search bowtie2  
$ conda search -c bioconda bowtie2
```

# Basic commands

## Create environment

```
$ conda create -n [env_name]  
$ conda activate [env_name]  
$ conda deactivate [env_name]
```

## list environment or packages

```
$ conda env list  
$ conda list
```

## Search tools

```
$ conda search bowtie2  
$ conda search -c bioconda bowtie2
```

## Install tools

```
$ conda install bowtie2  
$ conda install -c bioconda bowtie2  
$ conda install -c bioconda bowtie2=2.4.5
```

# Basic commands

## Create environment

```
$ conda create -n [env_name]  
$ conda activate [env_name]  
$ conda deactivate [env_name]
```

## list environment or packages

```
$ conda env list  
$ conda list
```

## Search tools

```
$ conda search bowtie2  
$ conda search -c bioconda bowtie2
```

## Install tools

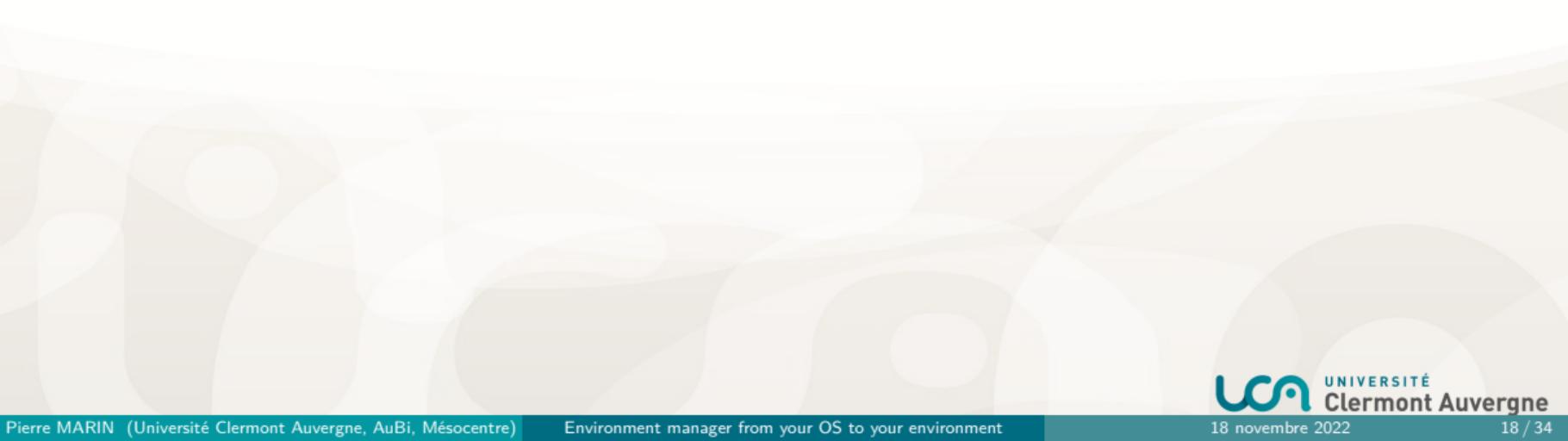
```
$ conda install bowtie2  
$ conda install -c bioconda bowtie2  
$ conda install -c bioconda bowtie2=2.4.5
```

## Remove tools

```
$ conda remove bowtie2
```

# Environment resolution

Conda also manage your environment to install compatible tools in the same environment



# Environment resolution

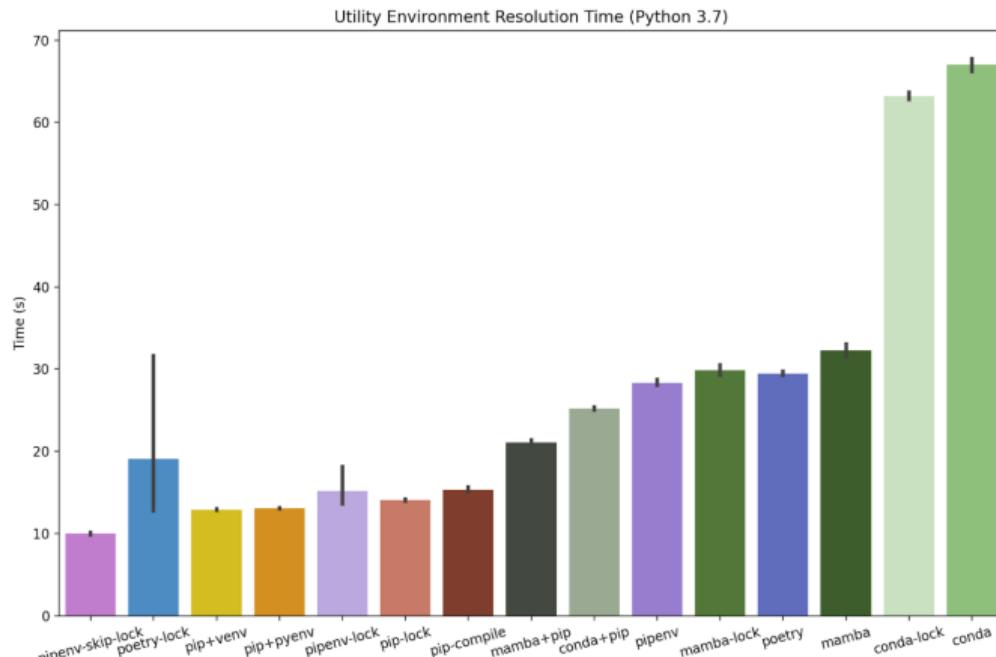
Conda also manage your environment to install compatible tools in the same environment

- time to solve the environment installation plan
- could be locked and don't install a tool

# Environment resolution

Conda also manage your environment to install compatible tools in the same environment

- time to solve the environment installation plan
- could be locked and don't install a tool



Container technology is not very old

The most famous : 

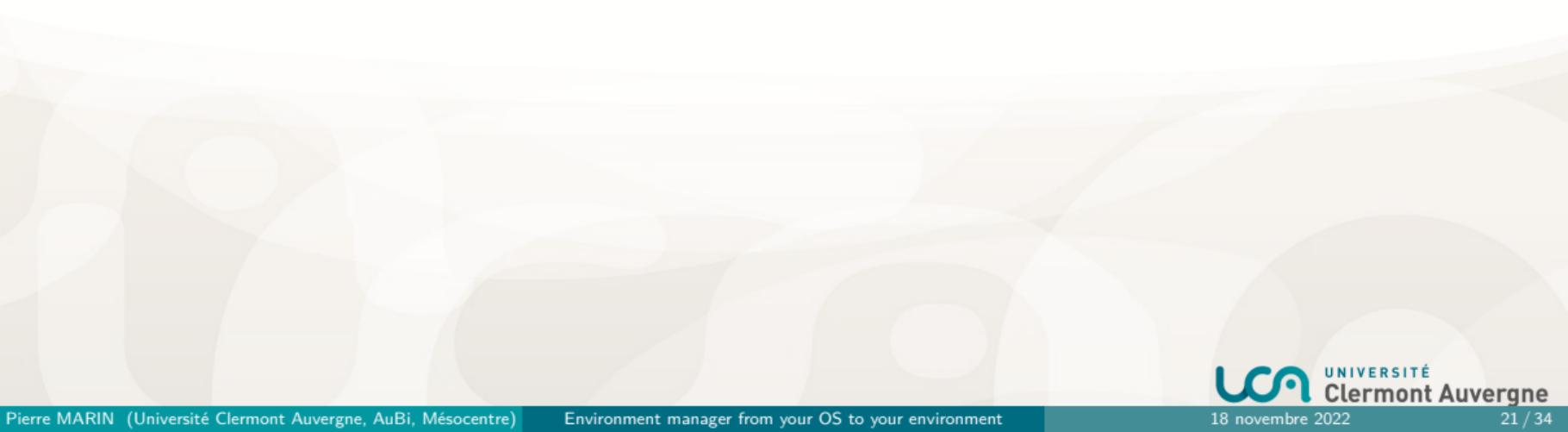
Solomon Hykes was inspired by container port in the world travel



Docker is an open source project, a community and a private company

- Born in 2010
- First public release in 2013
- V 1.0 in 2014
- Open source and free
- Packaged to Ubuntu in 2014 (V14.04)

# Term definitions



- Docker image -> "snapshot" immutable file

- Set of libraries, functions
- Static state
- Online Store or share
- Automatically build

- Docker image → "snapshot" immutable file

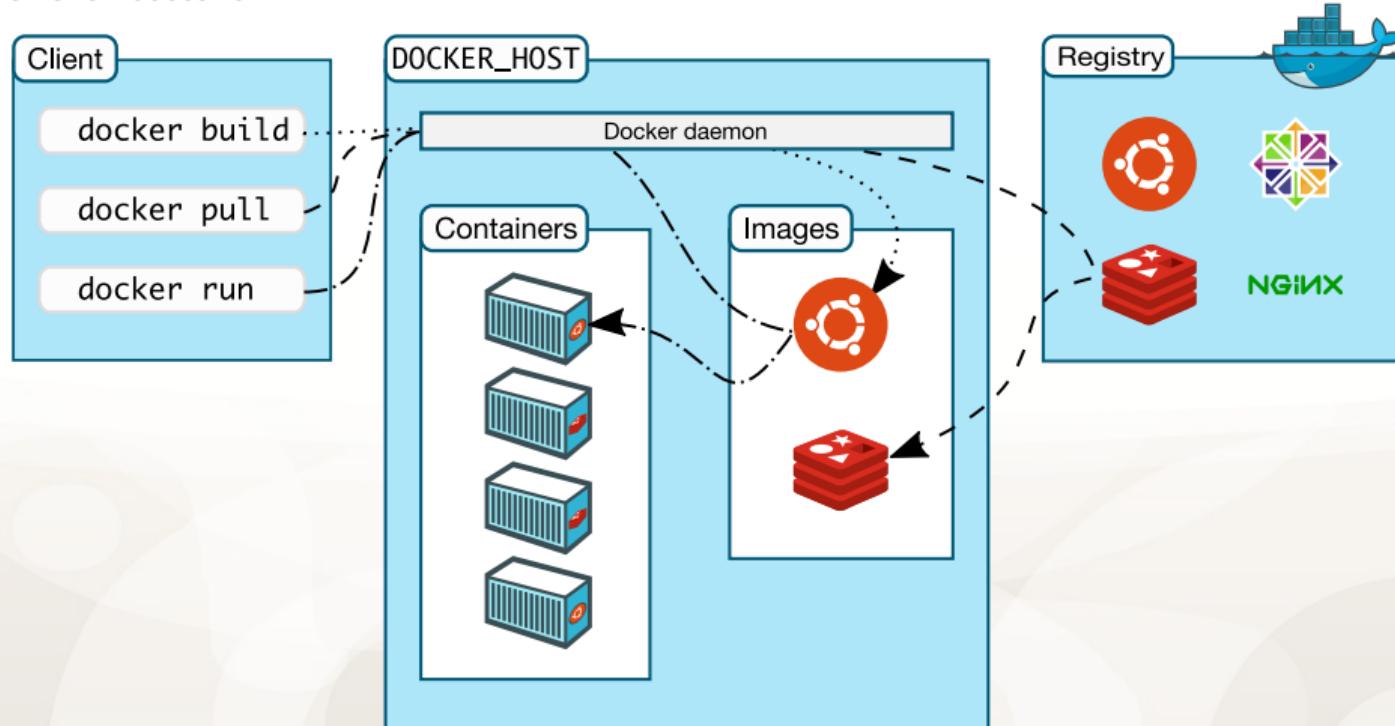
- Set of libraries, functions
- Static state
- Online Store or share
- Automatically build

- Docker container → instance of an image

- Result of the image activation
- Can be modified
- Can be turned into an image
- 1 image → multiple containers

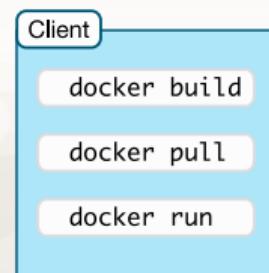
# Docker architecture

client-server architecture



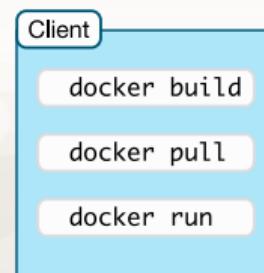
# Docker client

## 1 The user way to interact with Docker



# Docker client

- 1 The user way to interact with Docker
- 2 Client talk to a daemon (Docker background program)

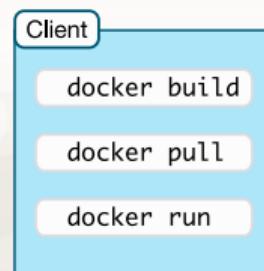


# Docker client

- 1 The user way to interact with Docker
- 2 Client talk to a daemon (Docker background program)
- 3 API based -> can communicate with several deamons

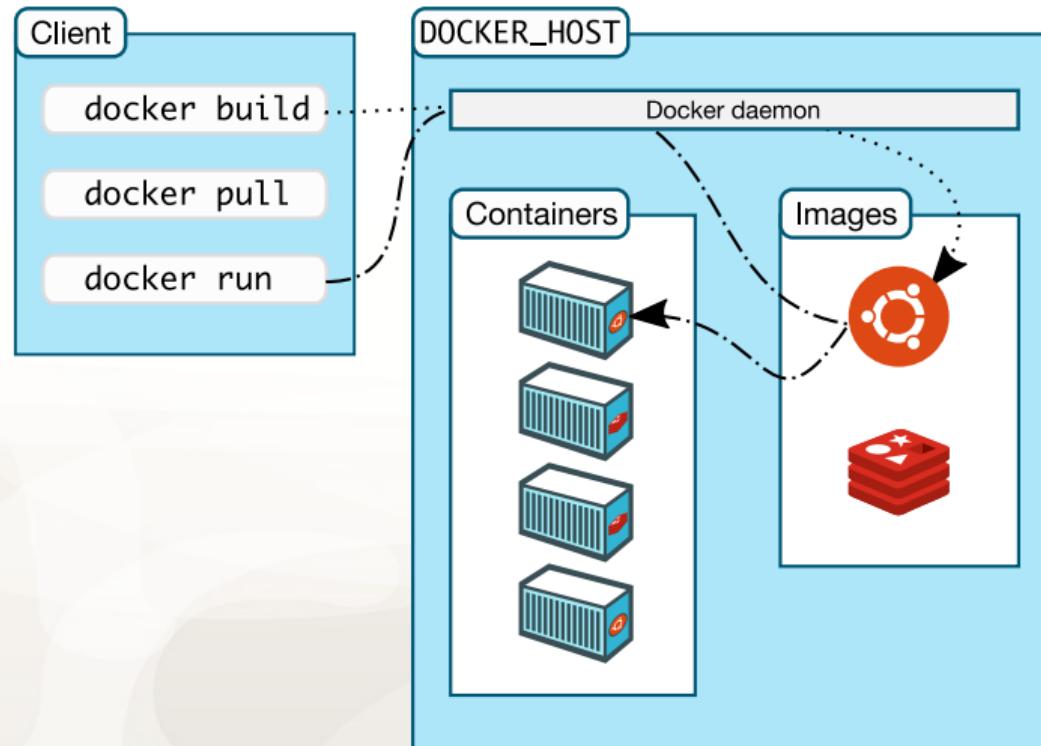
## Client

```
$ docker build [path] [url]  
docker build https://github.com/docker/rootfs.git#container:docker  
$ docker pull [image_name]  
docker pull biocontainers/samtools  
$ docker run [image_name]  
docker run biocontainers/samtools
```



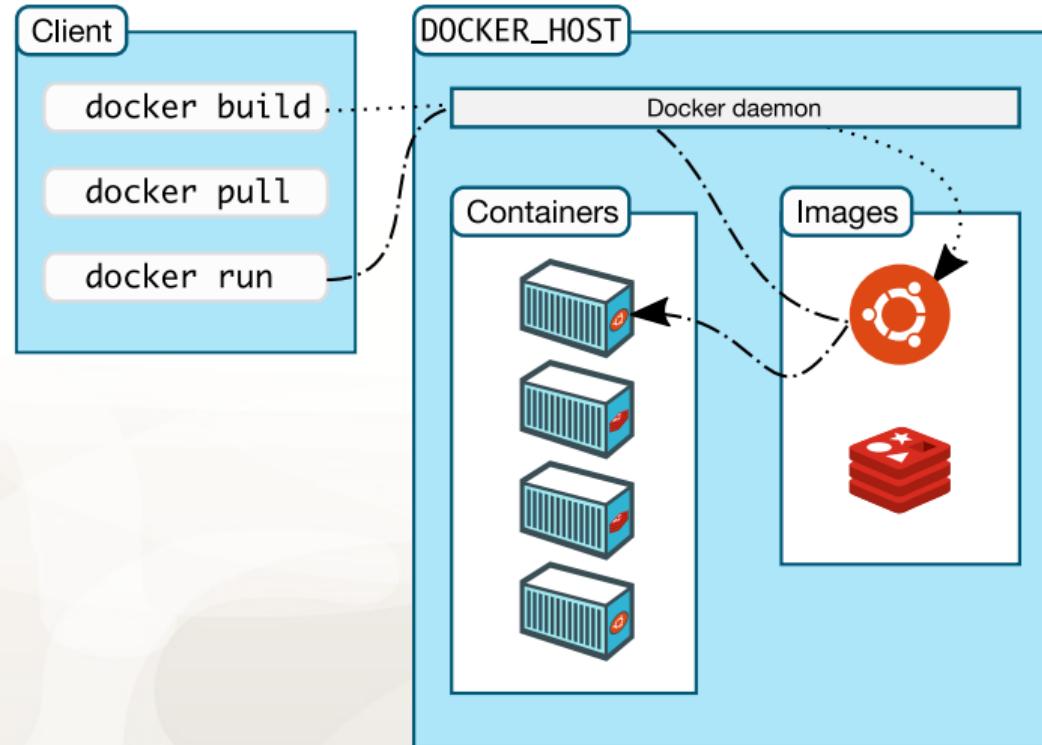
# Docker daemon

- 1 listen API requests



# Docker daemon

- 1 listen API requests
- 2 manage Docker images, containers...



# Docker registries

## 1 Store Docker images

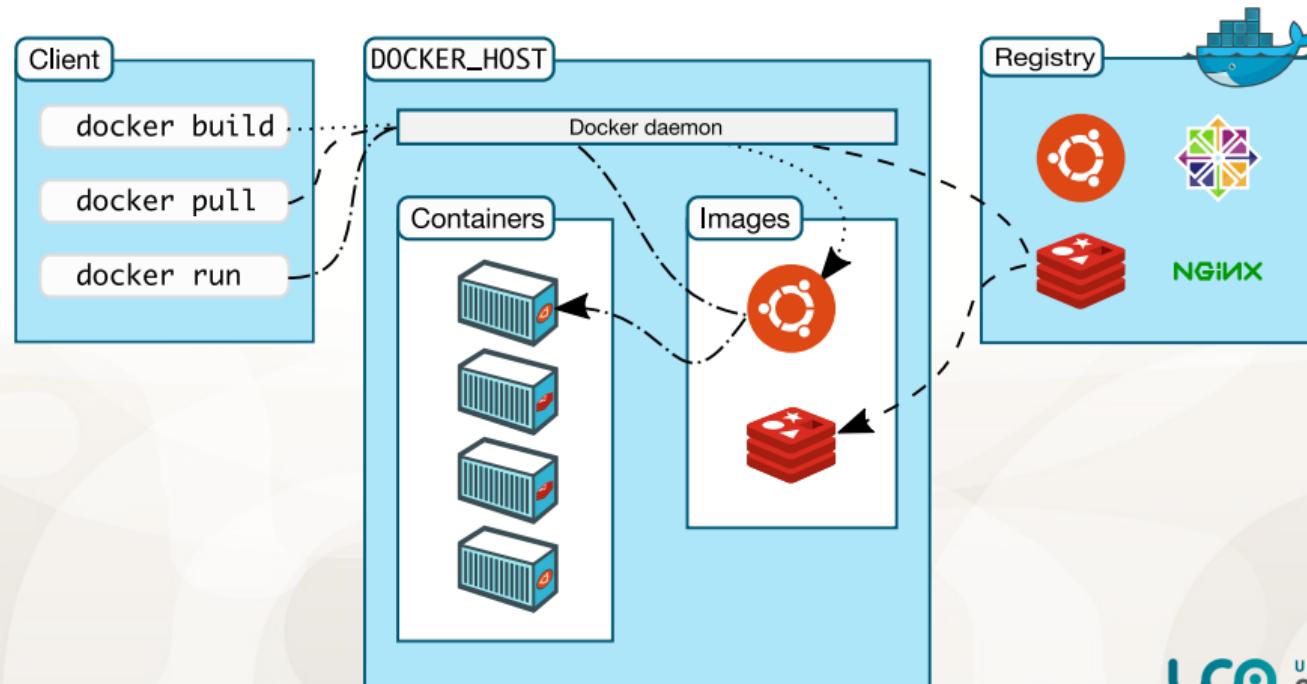
The screenshot shows the Docker Hub search interface for the query "rstudio". The search bar at the top contains "rstudio". Below it, there are navigation links for "Explore", "Pricing", "Sign In", and "Register". On the left, there are filters for "Images", "Extensions", "Plugins", "Docker Official Image", "Verified Publisher", and "Sponsored OSS". There are also filters for "Operating Systems" (Linux, Windows), "Architectures" (ARM, ARM 64, IBM POWER, IBM Z, PowerPC 64 LE), and "Best Match" dropdown.

1 - 25 of 2,061 results for **rstudio**.

Image	Publisher	Last Updated	Downloads	Stars
	ibmcom/rstudio-ppc64le	By IBM • Updated 3 years ago	487	4
	wholetale/rstudio-base	By wholetale • Updated 4 years ago	127	0
	bioconductor/rstudio_yescods	By bioconductor • Updated 16 days ago	23	0
	truecharts/rstudio	By truecharts • Updated 11 days ago	5	0

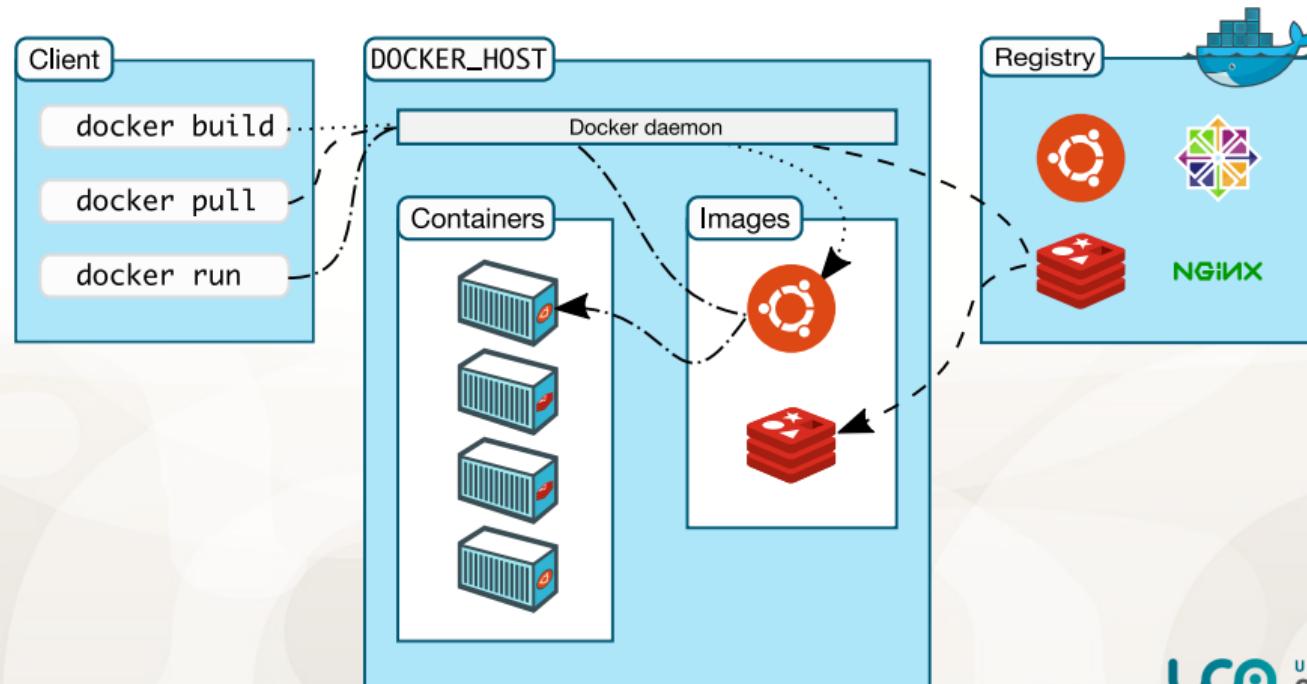
# Docker registries

- 1 Store Docker images
- 2 Docker hub is a public registry



# Docker registries

- 1 Store Docker images
- 2 Docker hub is a public registry
- 3 You can run your own registry



# Image layers

Focus on image building

- Layers building

```
$ docker pull debian
Using default tag: latest
latest: Pulling from library/debian
fdd5d7827f33: Pull complete
a3ed95caeb02: Pull complete
Digest: sha256:e7d38b3517548a1c71e41bffe9c8ae6d6d29546ce46bf62159837aad072c90aa
Status: Downloaded newer image for debian:latest
```

# Image layers

Focus on image building

- Layers building
- Several layers to one image

```
$ docker pull debian
Using default tag: latest
latest: Pulling from library/debian
fdd5d7827f33: Pull complete
a3ed95caeb02: Pull complete
Digest: sha256:e7d38b3517548a1c71e41bffe9c8ae6d6d29546ce46bf62159837aad072c90aa
Status: Downloaded newer image for debian:latest
```

# Image layers

Focus on image building

- Layers building
- Several layers to one image
- Some layers shared by images when pulling

```
$ docker pull debian
Using default tag: latest
latest: Pulling from library/debian
fdd5d7827f33: Pull complete
a3ed95caeb02: Pull complete
Digest: sha256:e7d38b3517548a1c71e41bffe9c8ae6d6d29546ce46bf62159837aad072c90aa
Status: Downloaded newer image for debian:latest
```

# Image layers

Focus on image building

- Layers building
- Several layers to one image
- Some layers shared by images when pulling
- Lighten the download and use of image on your computer

```
$ docker pull debian
Using default tag: latest
latest: Pulling from library/debian
fdd5d7827f33: Pull complete
a3ed95caeb02: Pull complete
Digest: sha256:e7d38b3517548a1c71e41bffe9c8ae6d6d29546ce46bf62159837aad072c90aa
Status: Downloaded newer image for debian:latest
```

# Docker Cheat Sheet



## Build

Build an image from the Dockerfile in the current directory and tag the image

```
docker build -t myimage:1.0 .
```

List all images that are locally stored with the Docker Engine

```
docker image ls
```

Delete an image from the local image store

```
docker image rm alpine:3.4
```



## Share

Pull an image from a registry

```
docker pull myimage:1.0
```

Retag a local image with a new image name and tag

```
docker tag myimage:1.0 myrepo/myimage:2.0
```

Push an image to a registry

```
docker push myrepo/myimage:2.0
```



## Run

Run a container from the Alpine version 3.9 image, name the running container "web" and expose port 5000 externally, mapped to port 80 inside the container.

```
docker container run --name web -p 5000:80 alpine:3.9
```

Stop a running container through SIGTERM

```
docker container stop web
```

Stop a running container through SIGKILL

List the running containers (add `--all` to include stopped containers)

```
docker container ls
```

Delete all running and stopped containers

```
docker container rm -f $(docker ps -aq)
```

Print the last 100

lines of a container's logs

```
docker container logs --tail 100 web
```



## Docker Management

All commands below are called as options to the base `docker` command. Run `docker <command> --help` for more information on a particular command.

<code>app*</code>	<i>Docker Application</i>
<code>assemble*</code>	<i>Framework-aware builds (Docker Enterprise)</i>
<code>builder</code>	<i>Manage builds</i>
<code>cluster</code>	<i>Manage Docker clusters (Docker Enterprise)</i>
<code>config</code>	<i>Manage Docker configs</i>
<code>context</code>	<i>Manage contexts</i>
<code>engine</code>	<i>Manage the docker Engine</i>
<code>image</code>	<i>Manage images</i>
<code>network</code>	<i>Manage networks</i>
<code>node</code>	<i>Manage Swarm nodes</i>
<code>plugin</code>	<i>Manage plugins</i>
<code>registry*</code>	<i>Manage Docker registries</i>
<code>secret</code>	<i>Manage Docker secrets</i>
<code>service</code>	<i>Manage services</i>
<code>stack</code>	<i>Manage Docker stacks</i>
<code>swarm</code>	<i>Manage swarm</i>
<code>system</code>	<i>Manage Docker</i>

# DOCKER COMPOSE CHEAT SHEET

## File

### structure

```
services:
  container1:
    properties: values

  container2:
    properties: values
```

```
networks:
  network:
```

```
volumes:
  volume:
```

## Types

### value

```
key: value
```

### array

```
key:
  - value
  - value
```

### dictionary

```
master:
  key: value
  key: value
```

## Properties

### build

```
build image from dockerfile
in specified directory
```

### container:

```
  build: ./path
  image: image-name
```

### image

```
use specified image
```

### container\_name

```
define container name to access
it later
```

```
container_name: name
```

### volumes

```
define container volumes to
persist data
```

### volumes:

```
  - /path:/path
```

### command

```
override start command for the
container
```

```
command: execute
```

### environment

```
define env variables for the
container
```

### environment:

```
  KEY: VALUE
  ...
  environment:
    - KEY=VALUE
```

### env\_file

```
define a env file for the
container to set and override
env variables
```

### env\_file:

```
  .env
  ...
  env_file:
    - .env
```

### restart

```
define restart rule
(no, always, on-failure, unless-
stopped)
```

### expose:

```
  - "9999"
```

### networks

```
define all networks for the
container
```

### networks:

```
  - network-name
```

### ports

```
define ports to expose to other
containers and host
```

### ports:

```
  - "9999:9999"
```

### expose

```
define ports to expose only to
other containers
```

### expose:

```
  - "9999"
```

### network\_mode

```
define network driver
(bridge, host, none, etc.)
```

### network\_mode:

```
host
```

### depends\_on

```
define build, start and stop
order of container
```

### depends\_on:

```
  - container-name
```

## Other

### idle container

```
send container to idle state
> container will not stop
```

```
command: tail -f /dev/null
```

### named volumes

```
create volumes that can be used in
the volumes property
```

### services:

```
  container:
    image: image-name
    volumes:
      - data-
    volume:/path/to/dir
```

### volumes:

```
  data-volume:
```

### networks

```
create networks that can be used
in the networks property
```

### networks:

```
  frontend:
    driver: bridge
```



# Singularity history

- Also a container manager as Docker



# Singularity history

- Also a container manager as Docker
- Open-source project

# Singularity history

- Also a container manager as Docker
- Open-source project
- Release in 2015

# Singularity history

- Also a container manager as Docker
- Open-source project
- Release in 2015
- More adopted in Academic institutions

# Singularity history

- Also a container manager as Docker
- Open-source project
- Release in 2015
- More adopted in Academic institutions
- Fork project in 2020 with now AppTainer (linux foundation) and SingularityCE

# Singularity history

- Also a container manager as Docker
- Open-source project
- Release in 2015
- More adopted in Academic institutions
- Fork project in 2020 with now AppTainer (linux foundation) and SingularityCE
- HPC compatible, no root whrite, integrate ressource managers (slurm)

# Singularity history

- Also a container manager as Docker
- Open-source project
- Release in 2015
- More adopted in Academic institutions
- Fork project in 2020 with now AppTainer (linux foundation) and SingularityCE
- HPC compatible, no root whrite, integrate ressource managers (slurm)
- Could use Docker images

# Singularity commands

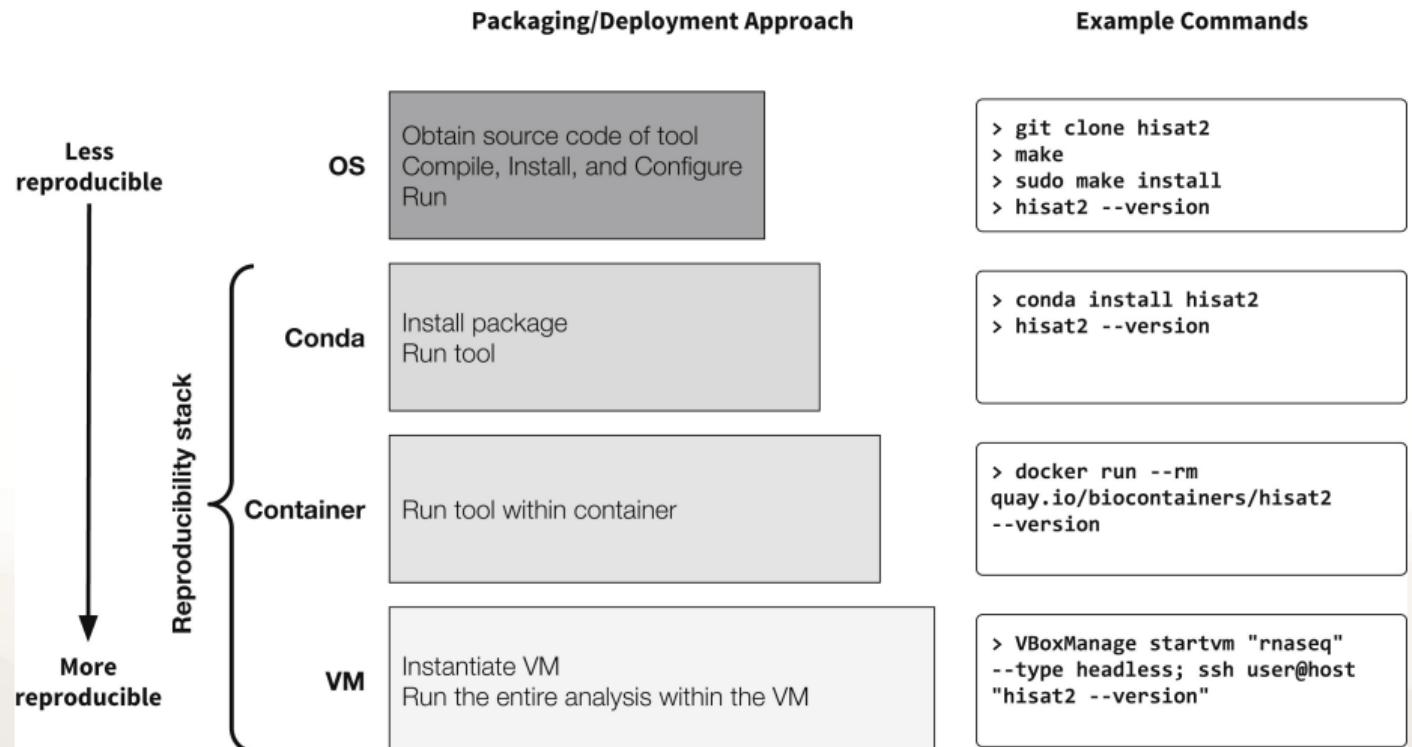
## classical commands

```
$ singularity search [image_name]  
$ singularity pull [image_name]  
$ singularity run [image_name]
```

# Singularity and Docker

## Singularity can use Docker images

```
$ singularity pull docker://debian:latest
INFO:    Converting OCI blobs to SIF format
INFO:    Starting build...
Getting image source signatures
Copying blob f606d8928ed3 done
Copying config 0311b76201 done
Writing manifest to image destination
Storing signatures
2022/10/06 10:50:41  info unpack layer: sha256:f606d8928ed378229f2460b94b504cca239fb9
INFO:    Creating SIF file...
```



1. Practical Computational Reproducibility in the Life Sciences Grüning et al, Cell Systems, 2018. DOI 10.1101/j.cels.2018.03.014

## Some recommandations

- A package first

. Recommendations for the packaging and containerizing of bioinformatics software Gruening, F1000 Research, 2019.  
DOI 10.12688/f1000research.15140.2



## Some recommandations

- A package first
- One tool, one container

. Recommendations for the packaging and containerizing of bioinformatics software Gruening, F1000 Research, 2019.  
DOI 10.12688/f1000research.15140.2



## Some recommandations

- A package first
- One tool, one container
- Tool and container versions should be explicit

. Recommendations for the packaging and containerizing of bioinformatics software Gruening, F1000 Research, 2019.  
DOI 10.12688/f1000research.15140.2



## Some recommandations

- A package first
- One tool, one container
- Tool and container versions should be explicit
- Avoid using ENTRYPOINT

. Recommendations for the packaging and containerizing of bioinformatics software Gruening, F1000 Research, 2019.  
DOI 10.12688/f1000research.15140.2

## Some recommandations

- A package first
- One tool, one container
- Tool and container versions should be explicit
- Avoid using ENTRYPOINT
- Reduce the size of your container as much as possible

. Recommendations for the packaging and containerizing of bioinformatics software Gruening, F1000 Research, 2019.  
DOI 10.12688/f1000research.15140.2

## Some recommandations

- A package first
- One tool, one container
- Tool and container versions should be explicit
- Avoid using ENTRYPOINT
- Reduce the size of your container as much as possible
- Keep data outside of the container

. Recommendations for the packaging and containerizing of bioinformatics software Gruening, F1000 Research, 2019.  
DOI 10.12688/f1000research.15140.2

## Some recommandations

- A package first
- One tool, one container
- Tool and container versions should be explicit
- Avoid using ENTRYPOINT
- Reduce the size of your container as much as possible
- Keep data outside of the container
- Add functional testing logic

. Recommendations for the packaging and containerizing of bioinformatics software Gruening, F1000 Research, 2019.  
DOI 10.12688/f1000research.15140.2

## Some recommandations

- A package first
- One tool, one container
- Tool and container versions should be explicit
- Avoid using ENTRYPOINT
- Reduce the size of your container as much as possible
- Keep data outside of the container
- Add functional testing logic
- Check the license of the software

. Recommendations for the packaging and containerizing of bioinformatics software Gruening, F1000 Research, 2019.  
DOI 10.12688/f1000research.15140.2

## Some recommandations

- A package first
- One tool, one container
- Tool and container versions should be explicit
- Avoid using ENTRYPOINT
- Reduce the size of your container as much as possible
- Keep data outside of the container
- Add functional testing logic
- Check the license of the software
- Make your package or container discoverable

. Recommendations for the packaging and containerizing of bioinformatics software Gruening, F1000 Research, 2019.  
DOI 10.12688/f1000research.15140.2

## Some recommandations

- A package first
- One tool, one container
- Tool and container versions should be explicit
- Avoid using ENTRYPOINT
- Reduce the size of your container as much as possible
- Keep data outside of the container
- Add functional testing logic
- Check the license of the software
- Make your package or container discoverable
- Provide reproducible and documented builds

. Recommendations for the packaging and containerizing of bioinformatics software Gruening, F1000 Research, 2019.  
DOI 10.12688/f1000research.15140.2

## Some recommandations

- A package first
- One tool, one container
- Tool and container versions should be explicit
- Avoid using ENTRYPOINT
- Reduce the size of your container as much as possible
- Keep data outside of the container
- Add functional testing logic
- Check the license of the software
- Make your package or container discoverable
- Provide reproducible and documented builds
- Provide helpful usage message

. Recommendations for the packaging and containerizing of bioinformatics software Gruening, F1000 Research, 2019.  
DOI 10.12688/f1000research.15140.2

## Encapsulation PRACTICE

Conda and Singularity