

FAIR_bioinfo : Open Science and FAIR principles in a bioinformatics project

How to make a bioinformatics project more reproducible

C. Hernandez¹ T. Denecker² J. Sellier² G. Le Corguillé²
C. Toffano-Nioche¹

¹Institute for Integrative Biology of the Cell (I2BC)
UMR 9198, Université Paris-Sud, CNRS, CEA
91190 - Gif-sur-Yvette, France

²IFB Core Cluster taskforce

June 2021



General information

Practical information:

- Dates: June 28th - 30th
- Location: Institut des Systèmes Complexes, 113 rue Nationale, 75013-Paris
- Courses: 9:00 to 17:30
- Meal: 12:30-14:00
- Pauses: 10:30-11:00 + 15:30-16:00
- 2 days of courses + 1 day of course building

Round table:





- Teachers
- Learners

Ressources:








- 
- GitLab
- L^AT_EX

Training schedule

Day 1:

- Introduction to reproducibility
- History management (3 Practical Sessions,  git,  GitHub)
- Control your development environment (1 PS,  CONDA)
- Encapsulation (2 PS,  docker)

Day 2:

- Workflow (2 PS,  SNAKEMAKE)
- Traceability with notebooks (2 PS,  JUPYTER, 
- IFB resources (2 PS,  SLURM, 
- Sharing and disseminating ( GitHub,  zenodo)
- Conclusion

Day 3:

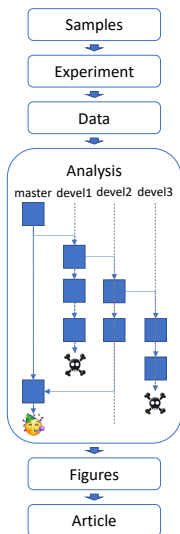
- Empowerment and improvement of resources

Table of contents

- 1 Introduction to reproducibility
- 2 History management
- 3 Control your development environment**
 - Introduction
 - Encapsulation
 - The Conda stack
 - What is Docker?
 - Practical session
- 4 Workflow
- 5 Tracability with Notebook
- 6 IFB resources
- 7 Sharing and dissemination
- 8 Conclusion
- 9 3rd Day

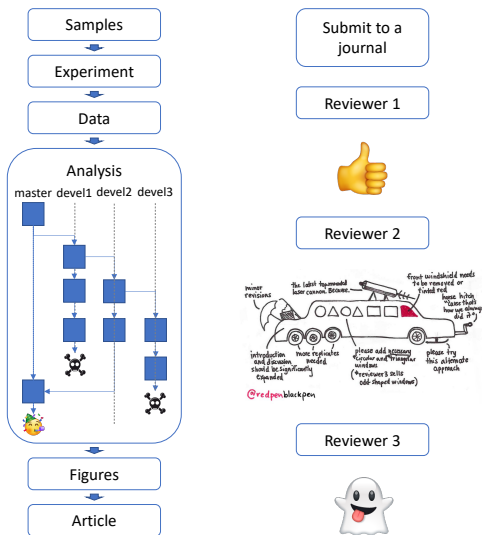
Introduction

A (not-so-uncommon) nightmare



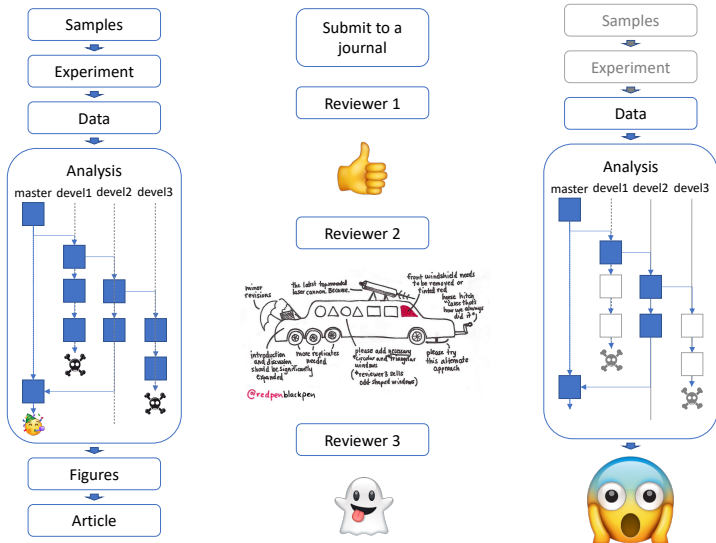
Introduction

A (not-so-uncommon) nightmare



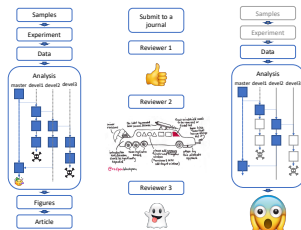
Introduction

A (not-so-uncommon) nightmare



Introduction

A (not-so-uncommon) nightmare






What changed?

- Package
- Software
- Libraries
- Environment variables
- OS version
- Computer
- ..?

Different levels of encapsulation

Goal : capture the system environment of applications (OS, packages, libraries,...) to control their execution.

- Hardware virtualisation (virtual machines) 
- OS virtualisation (images and containers) 
- Environment management  **CONDA**

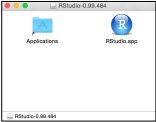
Encapsulation

Let's say we want to install RStudio...

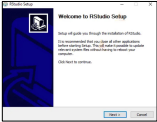
Install Rstudio ?



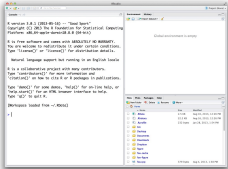
MacOS



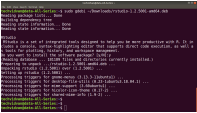
Windows



Use Rstudio



Unix-based



Encapsulation

We started with a computer using a specific OS...

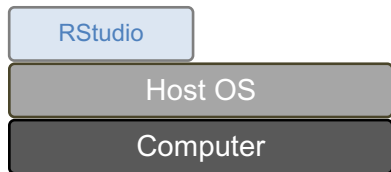


Host OS

The diagram consists of two stacked rectangular boxes. The top box is light gray and contains the text 'Host OS'. The bottom box is dark gray and contains the text 'Computer'. The 'Host OS' box is centered within the 'Computer' box, illustrating the concept of encapsulation where the OS is contained within the hardware.

Computer

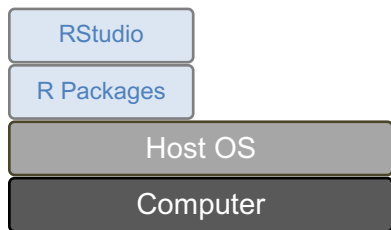
Encapsulation



We started with a computer using a specific OS...

And inside this environment, we installed a new application.

Encapsulation

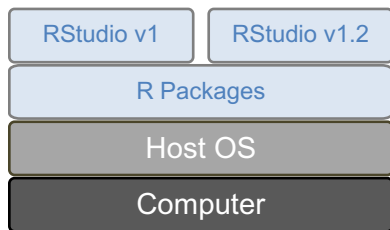


We started with a computer using a specific OS...

And inside this environment, we installed a new application.

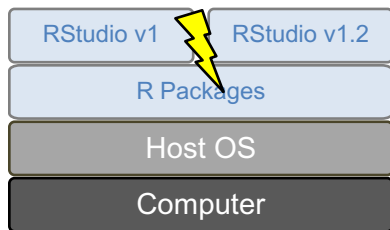
Applications rely on dependencies, e.g. external libraries.

Encapsulation



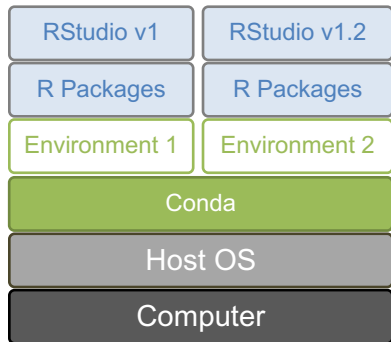
Usually dependencies of different applications don't interfere.
But what if we want to test the latest version of our favourite tool?
There might be conflicts. . .

Encapsulation



Usually dependencies of different applications don't interfere. But what if we want to test the latest version of our favourite tool? There might be conflicts. . .

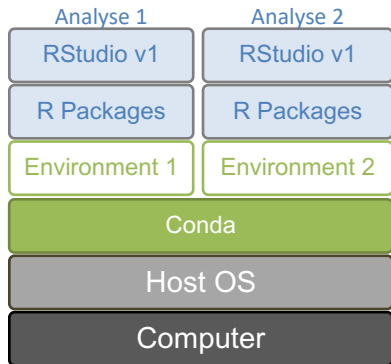
Encapsulation : managing environments



Idea : create separated environments for each application.

The logo for Conda, featuring a green circular icon with a white DNA helix on the left, followed by the word 'CONDA' in a bold, green, sans-serif font.

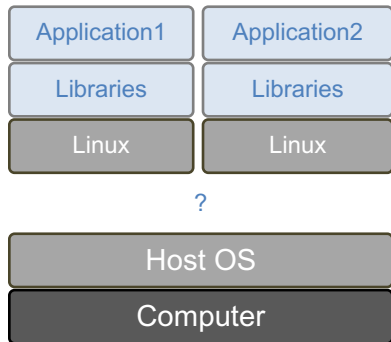
Encapsulation : managing environments



Idea : create separated environments for each application.

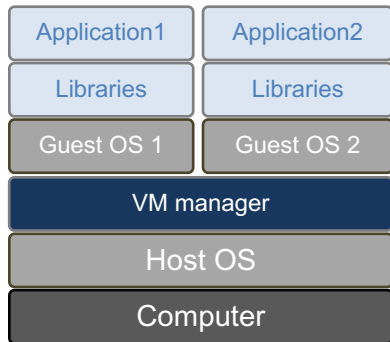
More versatile: create a new environment per analysis.

Encapsulation : hardware virtualisation



But what if we want to install a software from a different OS?

Encapsulation : hardware virtualisation

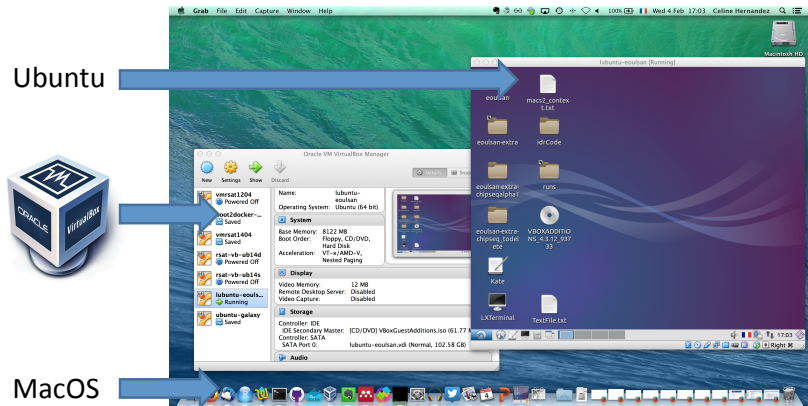


Idea: use virtual machines

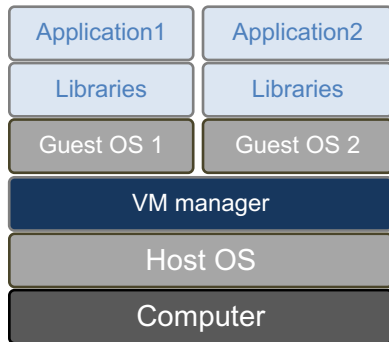
Pros:

- Each application gets a completely different and independent environment
- Virtual machines can be transferred to another computer (using the same manager)

Encapsulation : hardware virtualisation



Encapsulation : hardware virtualisation



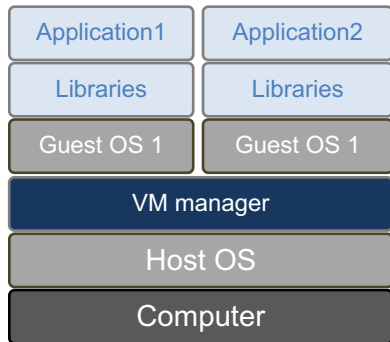
Idea: use virtual machines

Pros: transferable independent environments

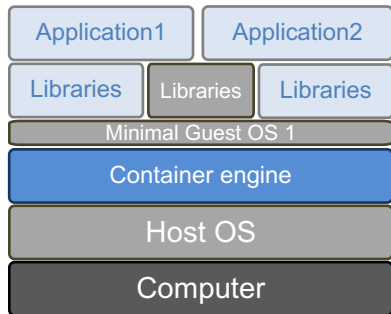
Cons:

- Redundancy between VMs
- Heavy to set up
- No automation

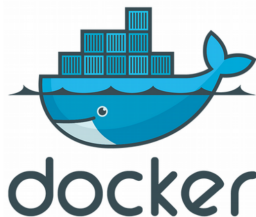
Encapsulation : OS virtualisation



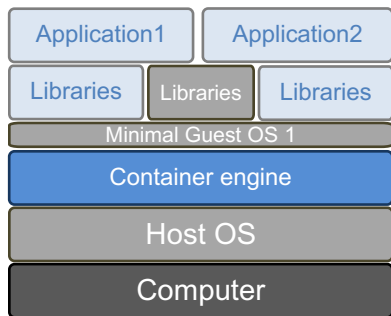
Encapsulation : OS virtualisation



Idea: "trick" applications into believing that they are in a different OS than the host's
Avoid redundancy.



Encapsulation : OS virtualisation

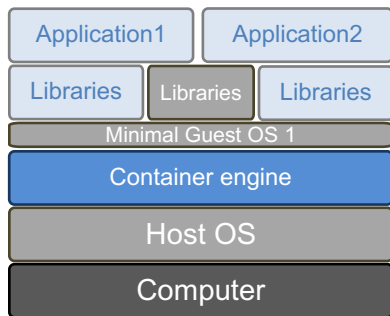


OS virtualisation vs hardware virtualisation

Pros:

- Speed
 - ▶ Installation is faster
 - ▶ No boot time
- Lightweight
 - ▶ Minimal base OS
 - ▶ Minimal libraries and application set
- Easy sharing of applications

Encapsulation : OS virtualisation



Cons:

- Singularity to use images on a cluster
- Changes of policies of the Docker company

Update of the Docker Image retention policy (13/08/2020)

What is a container image retention limit and how does it affect my account?

Image retention is based on the activity of each individual image stored within a user account. If an image has not either been pulled or pushed in the amount of time specified in your subscription plan, the image will be tagged "inactive." Any images that are tagged as "inactive" will be scheduled for deletion. Only accounts that are on the **Free** individual or organization plans will be subject to image retention limits. A new dashboard will also be available in Docker Hub that offers the ability to view the status of all of your container images.

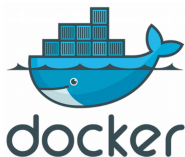
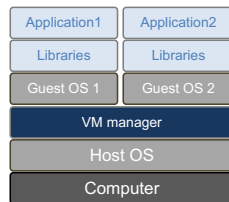
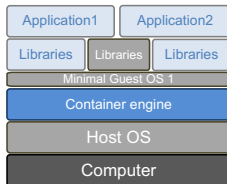
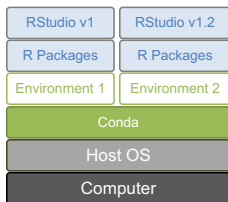
What are the new container image retention limits?

Docker is introducing a container image retention policy which will be enforced starting November 1, 2020. The container image retention policy will apply to the following plans:

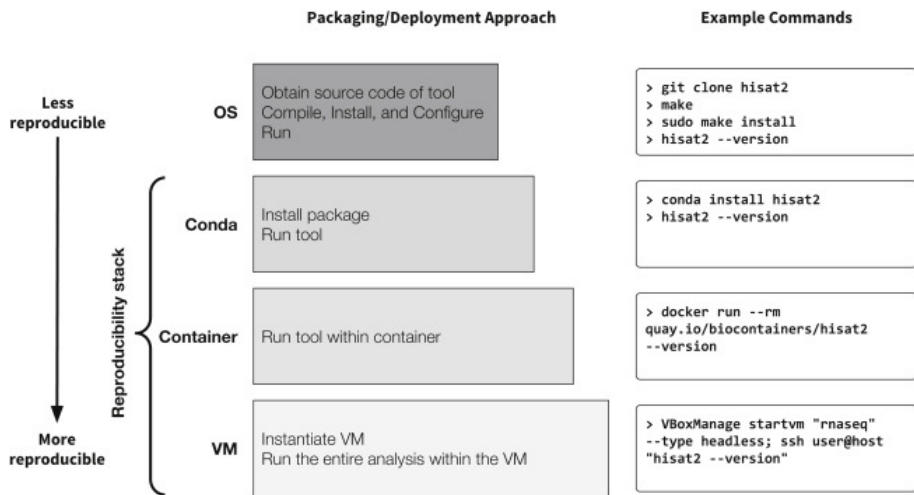
- Free plans will have a 6 month image retention limit
- Pro and Team plans will have unlimited image retention

<https://www.docker.com/pricing/retentionfaq>

Encapsulation



Encapsulation and reproducibility stack



Practical Computational Reproducibility in the Life Sciences - Björn Grüning et al (2018)



Conda

CONDA: an environment manager

Conda definitions

- Environment: a set of packages/tools in a directory (added to our PATH)
- Conda: an open source package + a general-purpose environment management system (installation, execution, upgrade). For any programming language, multi-platform (Windows, MacOS, Linux).
- Conda package: a compressed tarball of a tool

Why using an environment manager?

- avoid compilation and dependencies problems: an environment manager will take care of everything!
- have several environments in parallel each with their own set of tools
- useful when cross-tools dependencies are incompatible with each other



Conda distribution

- Anaconda: a data science platform, comes with a lot of packages
- Miniconda: come without installed packages

Anaconda cloud, the "conda hub"

- [Anaconda cloud](#) (private company) relies on the community of developers, concerns many domains (Machine Learning, Data Visualization, Dashboarding-web, Image Processing, Natural Language Processing, etc)
- Anaconda cloud: made up of channels/owners. Each channels contains one or more conda packages
- be careful when downloading any packages from an untrusted source, always inspect before installation

Some conda channels




- default
- conda-forge: many popular python packages (analogous to PyPI but with a unified, automated build infrastructure and more peer review of recipes)
- bioconda: bioinformaticians' contributions
- private

Channels list order

- when different channels have the same package \Rightarrow collisions
- collisions resolved following the order of your channels list \Rightarrow put supplemental channels at the bottom of your channel list

Conda and R

The R interpreter is included in the `r-essentials` package (200 r packages). Add `r-` before the regular `r` package name (eg. `r-ggplot2`)

⬇ Favorites	⬇ Downloads	⬇ Package (owner / package)	Platforms
18	304565	 <code>r / r-essentials</code> 3.6.0 Some essential packages for working with R	linux-32 linux-64 osx-64 win-32 win-64 conda
11	198513	 <code>skyblued / r-essentials</code> 3.5.1 Some essential packages for working with R	linux-64 osx-64 win-32 win-64 copy conda
1	66845	 <code>conda-forge / r-essentials</code> 4.1 Some essential packages for working with R. This was migrated from the 'r' channel.	linux-64 noarch osx-64 conda

Mamba

A fast drop-in alternative to conda, using `libsolv` for dependency resolution

```
conda install -c conda-forge mamba
```

Next, replace `conda` by `mamba` to use it

simple commands

```
1 conda create env -n myenv # creation of a conda environment
2 conda info --envs # list environments (* for the active one)
3 conda activate myenv # active the myenv environment
4 conda list # list packages (only in an active environment)
5 conda install package # installation of a tool/package
6 conda remove package # suppress the tool from the
   environment
7 conda env remove -n myenv # suppress the myenv environment
8 conda deactivate # inactivate the environment
```

miniconda3

With the miniconda3 distribution and by default, environments are installed in a miniconda3/envs/ repository

interactive

- create an environment
- activate the environment
- install some conda packages

configuration file

- list all conda packages in a configuration file (yaml or json format)
- create the environment based on the configuration file (option `-f`)
- activate the environment

reproducibility

- good practice: use a configuration file
- specify a precise version of a package:
`<channel>::<package>=<version>`

Conda Exercise

Conda setup

How to access conda?

- Conda is so used that it could even be installed by default to your machine. To test this: `conda --version`
- if not, may install it or got it by a docker image:

```
1 docker run -i -t -v ${PWD}:/data continuumio/miniconda3
```

- already activated on the IFB cluster (otherwise with module: `module load conda`)



How to access tools?

Manage Conda environment

① create the working environment:

```
1 conda create env -n myenv
```

② activate it:

```
1 conda activate myenv
```

③ if not yet done, install packages (specify the channel):

```
1 conda install -c bioconda bowtie2
```

④ work with the tools

⑤ quite the environment:

```
1 conda deactivate
```

Install snakemake with conda

Objective

Create a conda configuration file to install the snakemake tool.

Hint

- Search its channel in the Anaconda cloud web pages
- the "minimal" environment is sufficient

Install snakemake with conda

condaEnvSmk.yml

```
1 channels:  
2   - conda-forge  
3   - bioconda  
4   - main  
5 dependencies:  
6   - snakemake-minimal=6.5.0
```

run

```
1 conda env create -n condaEnvSmk -f condaEnvSmk.yml  
2 conda activate condaEnvSmk  
3 snakemake --version
```



Docker

What is Docker?

Docker is not very “old”

- First commit January 2013
- First version March 2013
- Version 1.0 in June 2014

But its adoption was fast

- Officially packaged in Ubuntu since 2014 (v14.04)

What is Docker?

Image



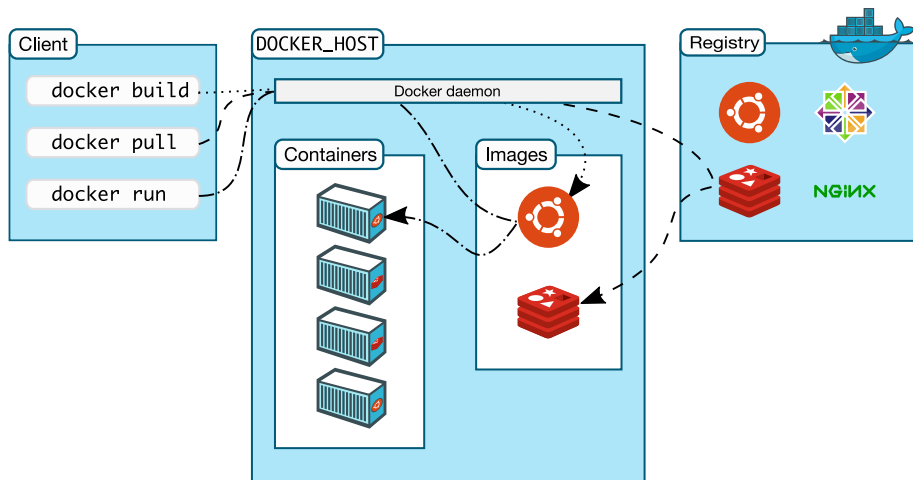
- Set of libraries and functions
- Fixed. Cannot be modified
- Can be stored/shared online
- Can be automatically built

Container



- "Active image"
- Can be modified (interactive)
- Can be turned into an image
- One image, many containers

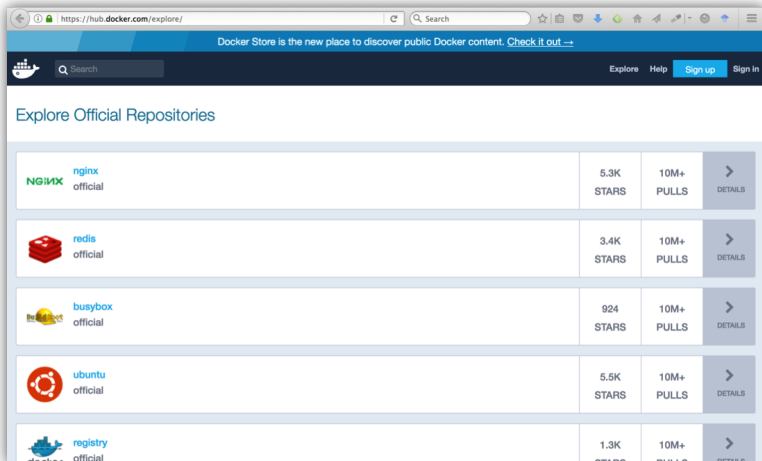
What is Docker?



(<https://docs.docker.com/get-started/overview/>)

What is Docker?

DockerHub



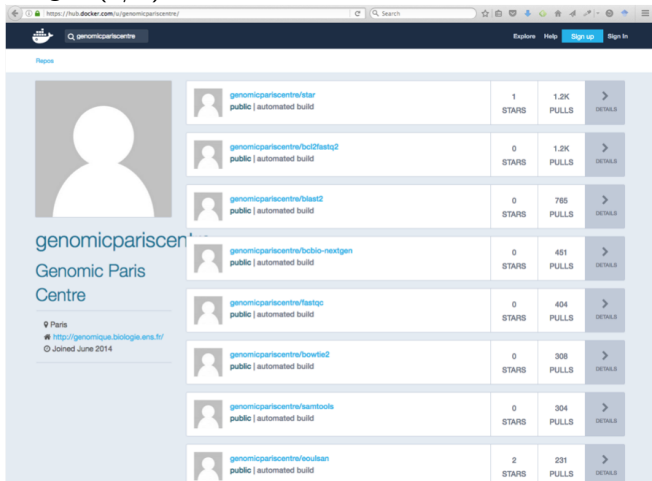
The screenshot shows the DockerHub website interface. At the top, there is a navigation bar with the text "Docker Store is the new place to discover public Docker content. Check it out →". Below this is a search bar and navigation links for "Explore", "Help", "Sign up", and "Sign in". The main content area is titled "Explore Official Repositories" and displays a list of five official Docker repositories in a table format.

Repository Name	Stars	Pulls	Action
nginx official	5.3K STARS	10M+ PULLS	DETAILS
redis official	3.4K STARS	10M+ PULLS	DETAILS
busybox official	924 STARS	10M+ PULLS	DETAILS
ubuntu official	5.5K STARS	10M+ PULLS	DETAILS
registry official	1.3K STARS	10M+ PULLS	DETAILS

(<https://hub.docker.com/>)

What is Docker?

Usermade images (1/2)



The screenshot shows the Docker Hub profile for 'genomicpariscentre'. The profile includes a bio: 'Genomic Paris Centre', location 'Paris', website 'http://genomique.biologie.ens.fr', and 'Joined June 2014'. Below the bio is a table of Docker images:

Image Name	Stars	Pulls	Details
genomicpariscentre/star public automated build	1	1.2K	>
genomicpariscentre/bol2fastq2 public automated build	0	1.2K	>
genomicpariscentre/blast2 public automated build	0	765	>
genomicpariscentre/bcbio-nextgen public automated build	0	451	>
genomicpariscentre/fastqc public automated build	0	404	>
genomicpariscentre/bowtie2 public automated build	0	308	>
genomicpariscentre/samtools public automated build	0	304	>
genomicpariscentre/eoulsan public automated build	2	231	>

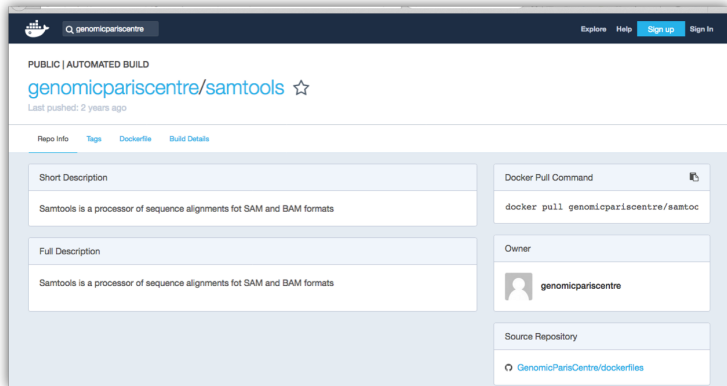
(url<https://hub.docker.com/u/genomicpariscentre/>)



What is Docker?

Usermade images (2/2)

Be critical!

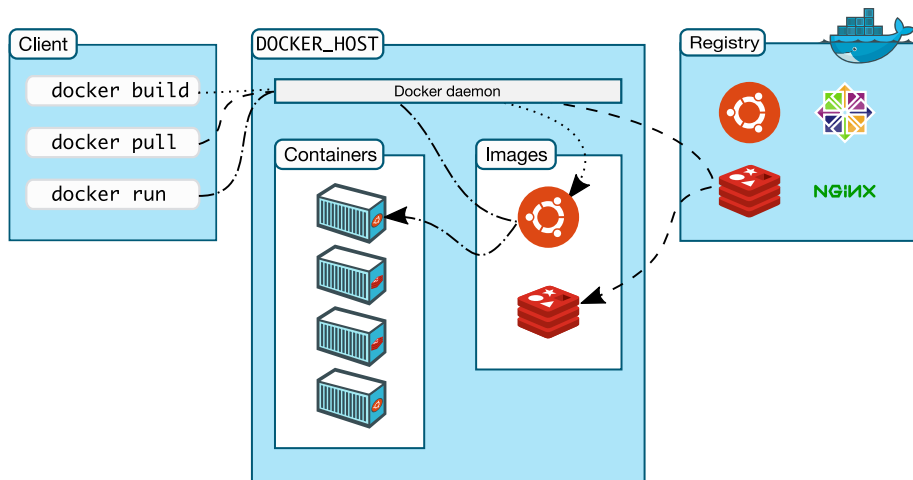


The screenshot shows the Docker Hub interface for the repository `genomicpariscentre/samtools`. The page is titled "PUBLIC | AUTOMATED BUILD" and shows the repository name with a star icon and the text "Last pushed: 2 years ago". Below the repository name are tabs for "Repo Info", "Tags", "Dockerfile", and "Build Details". The main content area is divided into two columns. The left column contains a "Short Description" and a "Full Description", both stating: "Samtools is a processor of sequence alignments for SAM and BAM formats". The right column contains a "Docker Pull Command" section with the command `docker pull genomicpariscentre/samtools`, an "Owner" section with a profile picture and the name "genomicpariscentre", and a "Source Repository" section with a link to "GenomicParisCentre/dockerfiles".

(<https://hub.docker.com/r/genomicpariscentre/samtools/>)



What is Docker?



(<https://docs.docker.com/get-started/overview/>)

What is Docker?

Other commands :

- docker images : list images available locally
- docker ps : status of containers
- docker rm : delete a container
- docker rmi : delete an image
- ...

(More details during the practical session.)