

Introduction to containerisation with docker

George Githinji

KEMRI-Wellcome Trust Research Programme, Kilifi, Kenya

NGS course March 2024

Containerization

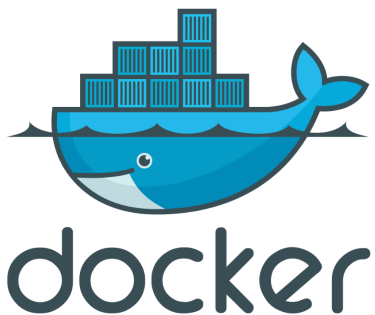


A software container is used to encapsulate a software component and the corresponding dependencies

- Sandbox

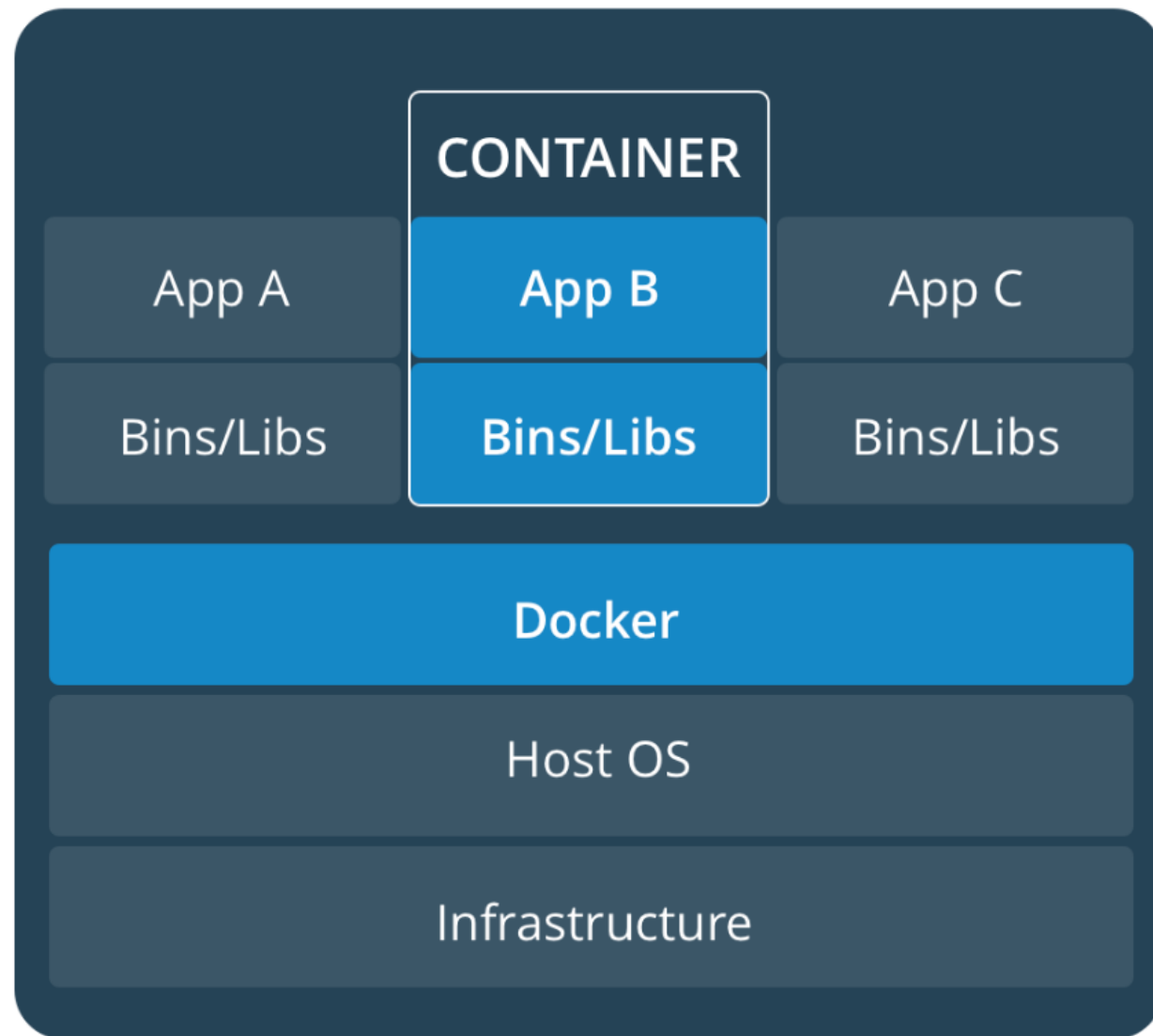
Containerisation

- A system that allows multiple isolated operating systems to run inside a larger, host system.
- The most basic type of containerization is **chroot**, which runs an application in a *jail* where it cannot see or access anything outside of its jail
- The most popular container technologies are Docker and Singularity



Docker

- An open platform for developing, shipping, and running applications.
- Docker separates your applications from your infrastructure



What are the benefits?

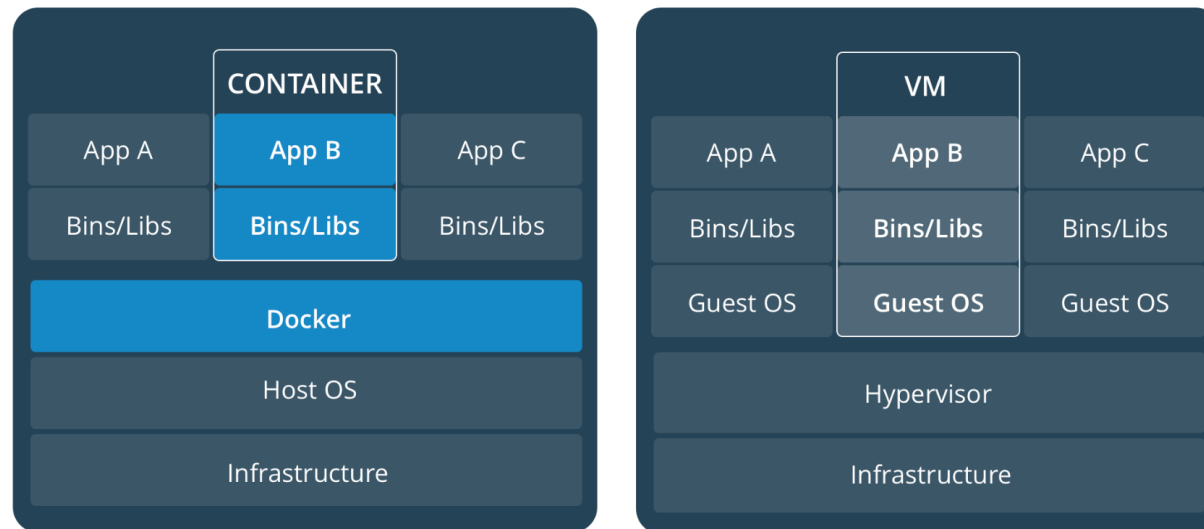
- Isolation
- Efficiency
- Scalability
- Portability
- Resource utilization
- Version control and Rollbacks
- Security
- Cost-effectiveness

Use-cases

- Analysis pipelines with many runtime tools (Python, Perl, etc) and many packages (Minimap, Samtools, GATK)
 - Snakemake, NextFlow,
- Web applications that need proxy server, databases and application code within a consistent operating environment
 - Galaxy.,

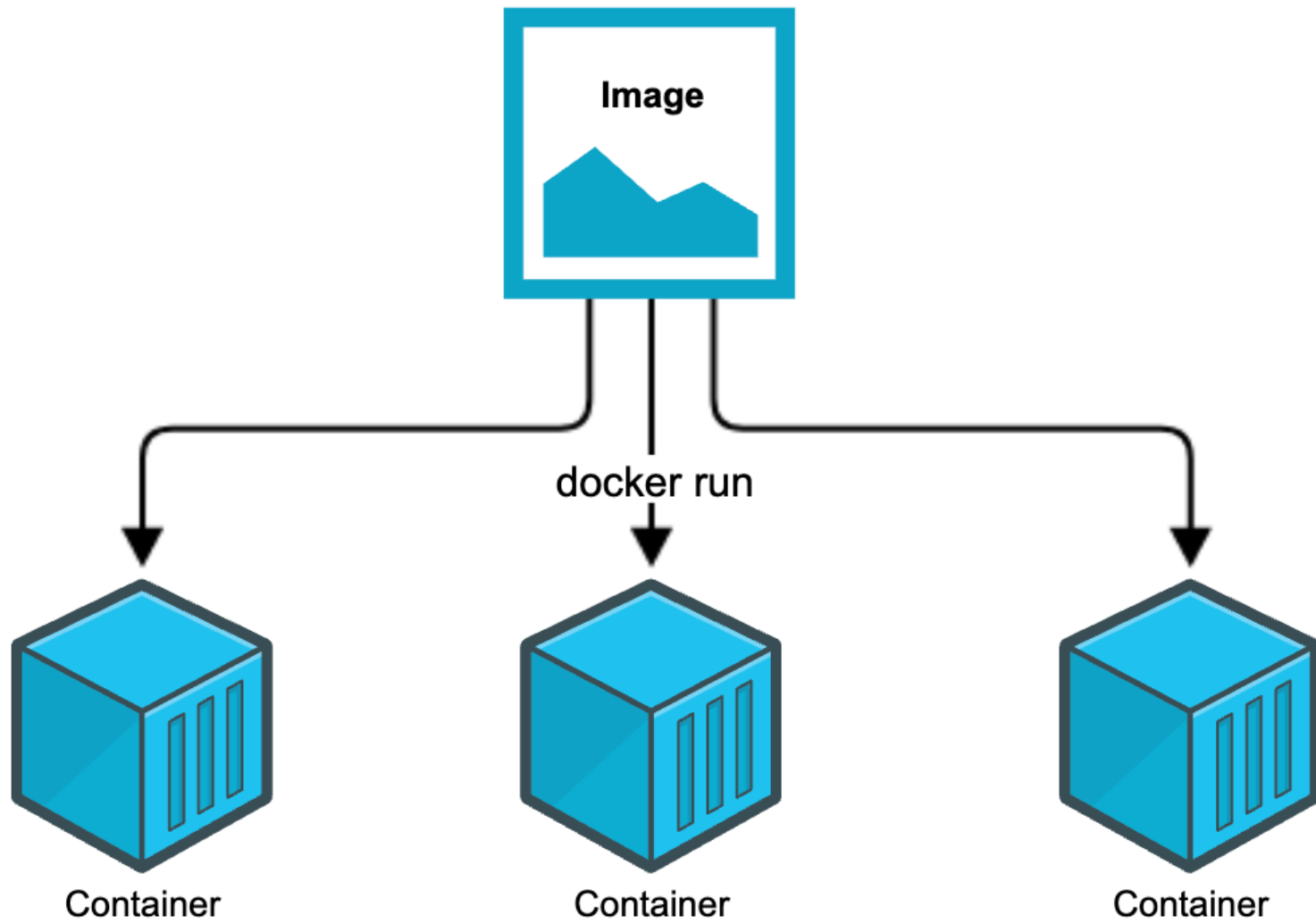
Docker or Virtual machines?

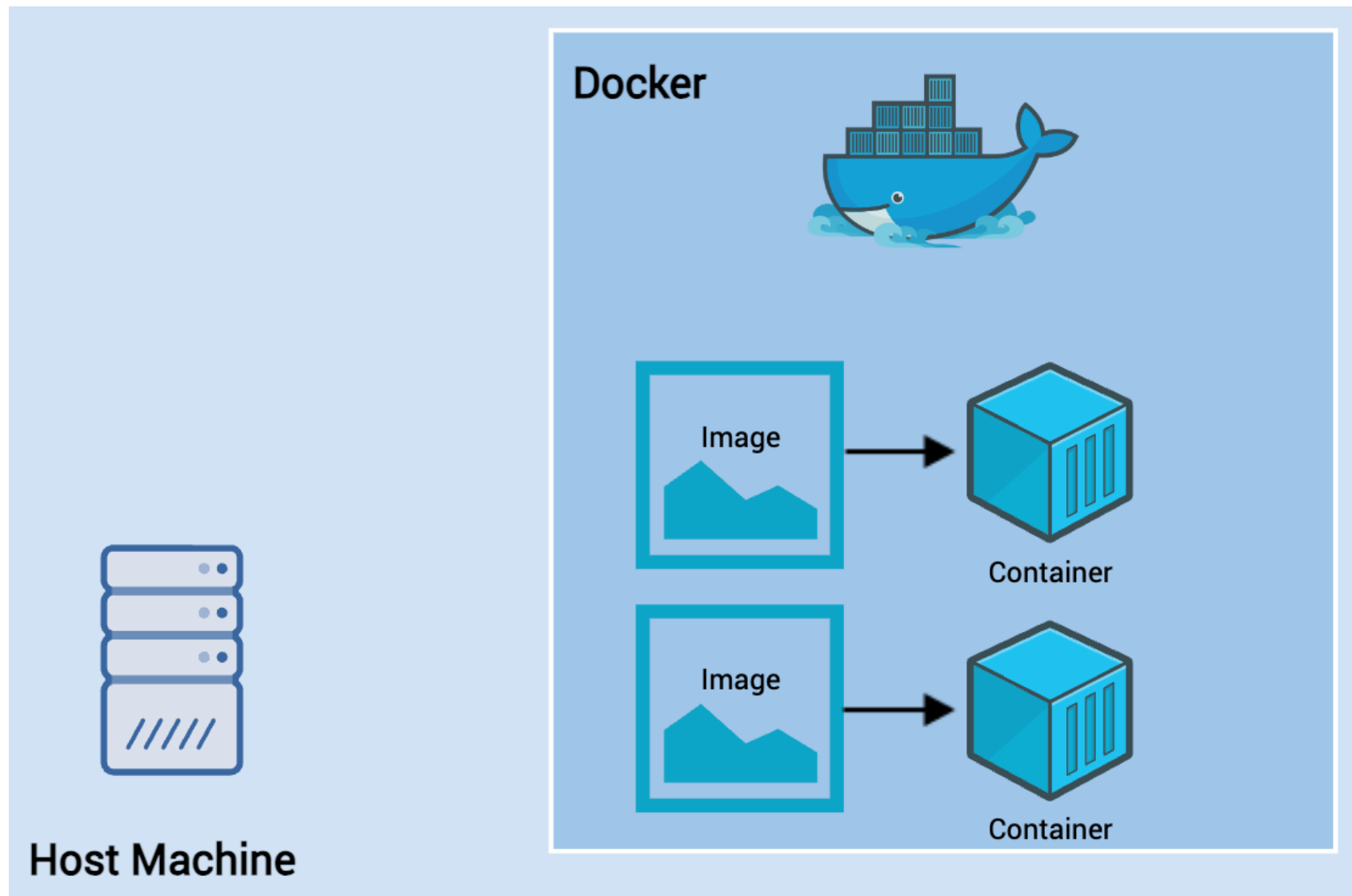
- Virtual are not lightweight
- Virtual machines package the entire guest OS.
- Docker uses the host **kernel and a minimal OS that can be shared between containers**



Terminology

- **Image** - *a lightweight, stand-alone, executable package of a piece of software that includes everything needed to run it: code, runtime, system tools, system libraries, settings*
- **Container** - *A running image*
- *Several containers can run the same image*
- **Host** – *the machine running docker on which images and containers are stored and running*





Installing docker

Instructions to get started with docker

<https://docs.docker.com/get-docker/>

Docker hub is a registry for docker containers

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

Running docker

- To run a **container**, you specify the **image** name to `docker run` command – docker will **pull** the image from **dockerhub**.

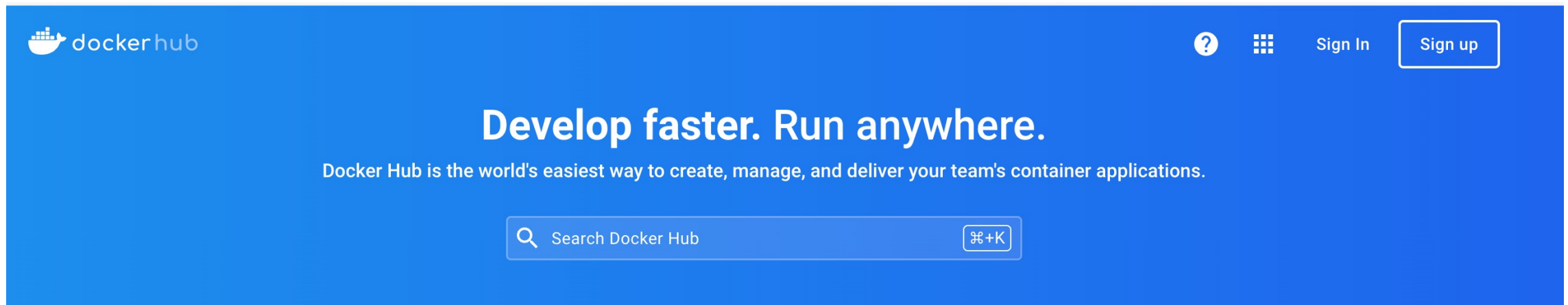
```
docker run image-name
```

```
→ ~ docker run hello-world
```

Docker hub

- A cloud registry of images

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



The image shows the Docker Hub homepage with a blue background. In the top left corner is the Docker Hub logo, which consists of a white ship icon and the text "dockerhub". In the top right corner, there are three icons: a question mark, a grid, and the text "Sign In". To the right of "Sign In" is a white button with a blue border and the text "Sign up". In the center of the page, the text "Develop faster. Run anywhere." is displayed in a large, bold, white font. Below this text is a smaller line of white text: "Docker Hub is the world's easiest way to create, manage, and deliver your team's container applications." At the bottom center, there is a white search bar with a magnifying glass icon on the left, the text "Search Docker Hub" in the middle, and a button with a blue border and the text "⌘+K" on the right.

dockerhub

?

Sign In

Sign up

Develop faster. Run anywhere.

Docker Hub is the world's easiest way to create, manage, and deliver your team's container applications.

Search Docker Hub

⌘+K

<https://biocontainers.pro/>

BioContainers Flow



Main Components

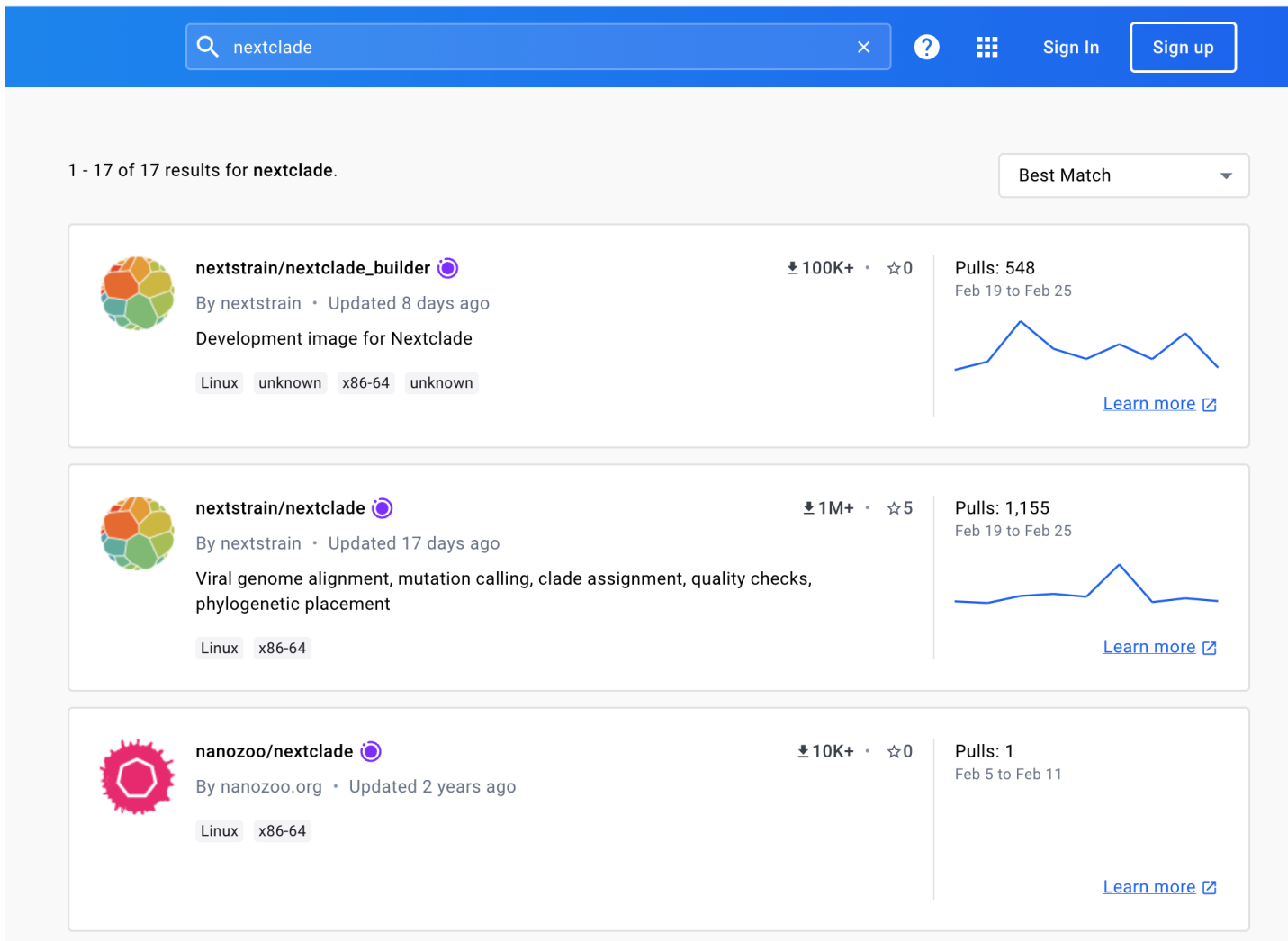
Registry

Specifications

Resources

Exercise 1: <https://hub.docker.com/>

Search for the best image for nextclade



The screenshot shows the Docker Hub search results for 'nextclade'. The search bar at the top contains 'nextclade' and the results are sorted by 'Best Match'. Three results are visible:

- nextstrain/nextclade_builder**: Development image for Nextclade. 100K+ downloads, 0 stars, 548 pulls. Updated 8 days ago. Supports Linux, unknown, x86-64, and unknown architectures.
- nextstrain/nextclade**: Viral genome alignment, mutation calling, clade assignment, quality checks, phylogenetic placement. 1M+ downloads, 5 stars, 1,155 pulls. Updated 17 days ago. Supports Linux and x86-64 architectures.
- nanozoo/nextclade**: 10K+ downloads, 0 stars, 1 pull. Updated 2 years ago. Supports Linux and x86-64 architectures.

Each result includes a 'Learn more' link and a pull count graph for the period of Feb 19 to Feb 25.


```
→ ~ docker pull nextstrain/nextclade
Using default tag: latest
latest: Pulling from nextstrain/nextclade
09e2bc8a597c: Pull complete
b3efbaa9bac1: Pull complete
9f669c487a27: Pull complete
Digest: sha256:2c2bd89cb129448d8ae4f196c390508819720e83f5856384919d3d2c59151130
Status: Downloaded newer image for nextstrain/nextclade:latest
docker.io/nextstrain/nextclade:latest
```

What's Next?

View a summary of image vulnerabilities and recommendations → [docker scout quickview nextstrain/nextclade](#)

Listing images

`docker images`

```
→ ~ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
nextstrain/nextclade	latest	c4e92a7ddab7	14 hours ago	147MB
nginx	latest	760b7cbba31e	2 weeks ago	192MB
ubuntu	latest	a50ab9f16797	3 weeks ago	69.2MB
alpine	latest	ace17d5d883e	5 weeks ago	7.73MB
hello-world	latest	ee301c921b8a	10 months ago	9.14kB

Docker run

Docker run [options] <image name> [image arguments]

Detached mode

Containers can run in the background, for example

```
docker run -d alpine
```

Docker prints out the ID of the container, so that you can access it later

Exercise: Run galaxy

- The image is called bgruening/galaxy-stable
- The galaxy image listens on port 80 inside the container
- Run the container in the background

```
docker run -d -p 80:80 bgruening/galaxy-stable
```

Port mapping

- Docker containers are free to listen on whatever ports to want to, for example port 80/443 for web requests

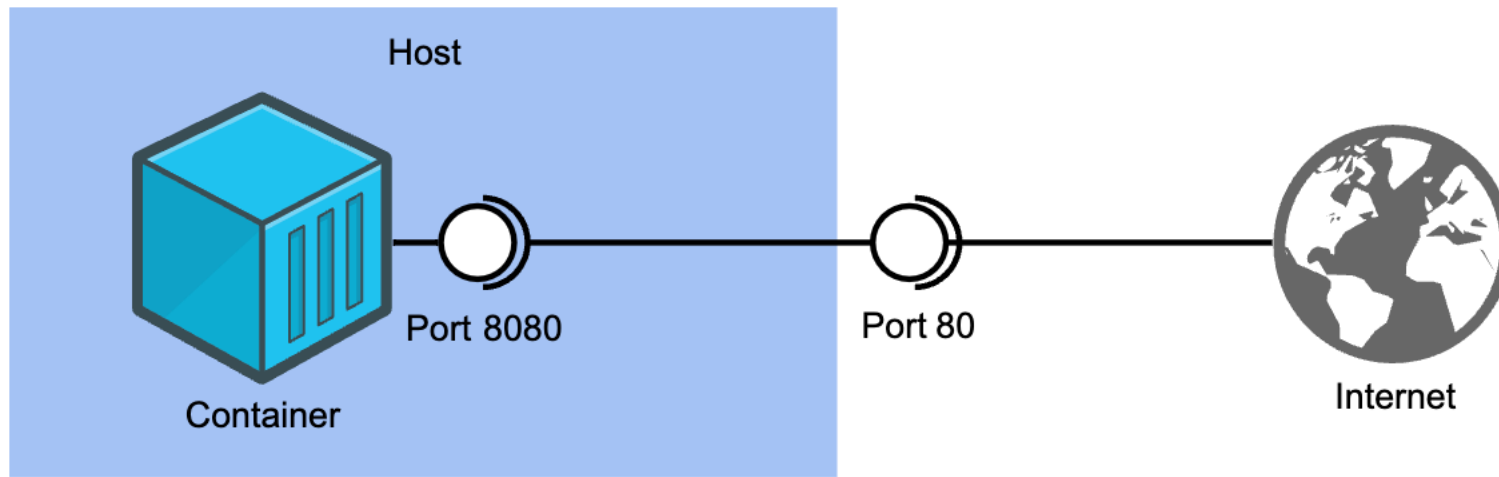
- However, these ports are not available on the host machine unless you use

```
docker run -p host:container <IMAGE NAME>
```

- This command means "map port 8080 inside this container to port 80 on the host machine

```
docker run -p 80:8080 <IMAGE NAME>
```

- Note that the host port and container port *can be the same*, and this is quite common



<http://localhost:80>

Listing running containers

List all running containers

```
docker ps
```

List terminated containers

```
docker ps -a
```

The IDs that are shown can be useful for other docker commands like

```
docker stop and docker exec
```


Docker stop

- Ordinarily, you can press ctrl+c to stop a container currently running in your terminal
- However, if the container is running in the background (with -d), or refuses to close, you can use docker stop

```
docker stop <CONTAINER ID>
```

- Use `docker stop` if you to close the currently running Galaxy Docker container
- Hint: use `docker ps` if you've forgotten the container's ID

Volumes and Bind Mounts

- By default, Docker containers cannot access data on the host system. This means
 - You can't use host data in your containers
 - All data stored in the container will be lost when the container exits
- You can solve this in two ways:
 - `-v /path/in/host:/path/in/container:` This **bind mounts** a host file or directory into the container. Writes to one will affect the other. Note that both paths have to be *absolute* paths, so you often want to use ``pwd` /some/path`
 - `-v volume_name:/path/in/container.` This mounts a **named volume** into the container, which will live separately from the rest of your files. This is preferred, unless you need to access or edit the files from the host

Exercise

- you need to store the logs for your Galaxy image on your host system using a bind mount
- The Galaxy container stores its logs in `/home/galaxy/logs`
- What command do you run?
- *Hint: You will want to run the container in detached mode*
- Once you have done this, Is the directory you mounted into the container to verify that you have the logs

```
docker run -d -p 80:80 -v `pwd`/galaxy_logs:/home/galaxy/logs bgruening/galaxy-  
stable
```

Running command inside a container

- You can run a command inside a running container using:

```
docker exec <CONTAINER ID> <COMMAND>
```

- For example:

```
docker exec bd2ac6cce96f ls
```

- You can also run an interactive bash session inside the container with:

```
docker exec -it bd2ac6cce96f bash
```

Start another Galaxy container using:

```
docker run -d -p 80:80 bgruening/galaxy-stable
```

Make a quick edit to the Galaxy homepage, which is located at `/etc/galaxy/web/welcome.html`

Edit the welcome message in some way, save the file, and then check to see if your changes worked on the website

Re-open the webpage in separate window or browser to get it to refresh

```
docker ps to find the container ID
```

```
docker exec -it <CONTAINER_ID> bash
```

Now, run `nano /etc/galaxy/web/welcome.html` **(or vim!)**
and save the file

Summary

`docker [-d] [-p host:container] [-v /host/path:/container/path] run <IMAGE NAME> - runs a Docker image`

`docker images` - displays all installed images

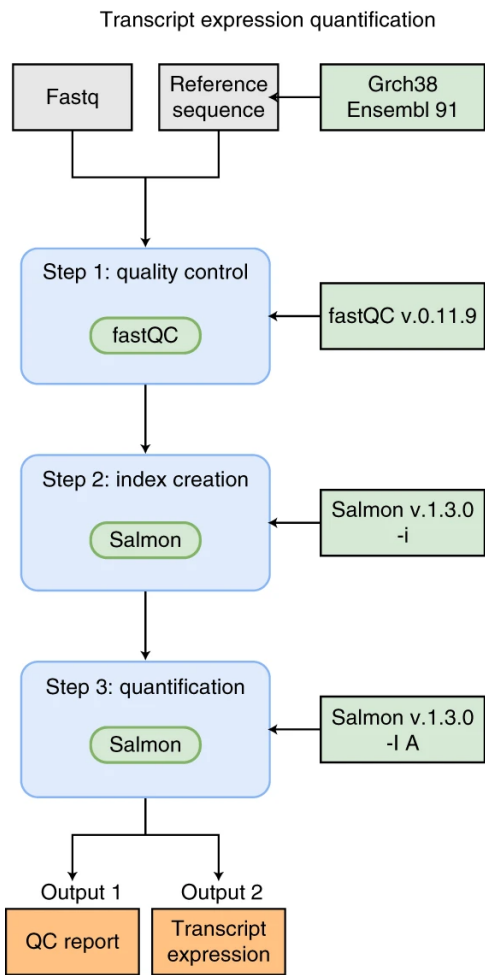
`docker ps [-a]` - displays all containers on the system

`docker exec <CONTAINER ID> <COMMAND>` - lets you run a command inside a running container

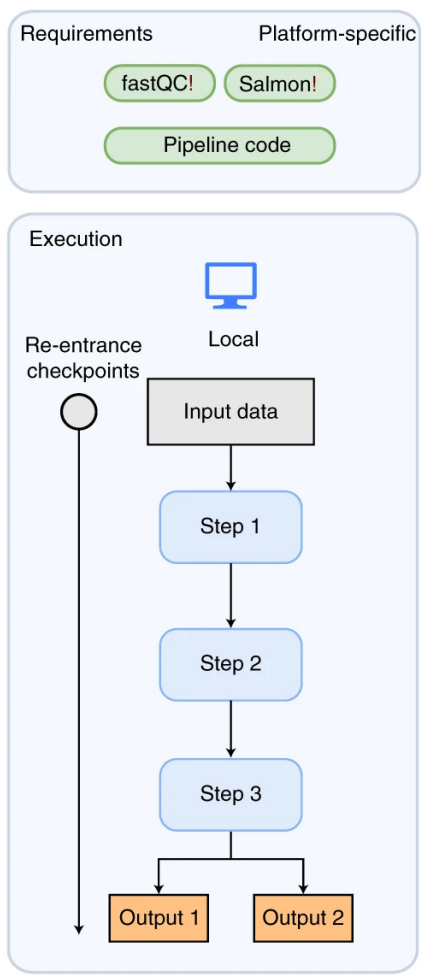
`docker stop <CONTAINER ID>` - stops a running container

Workflows

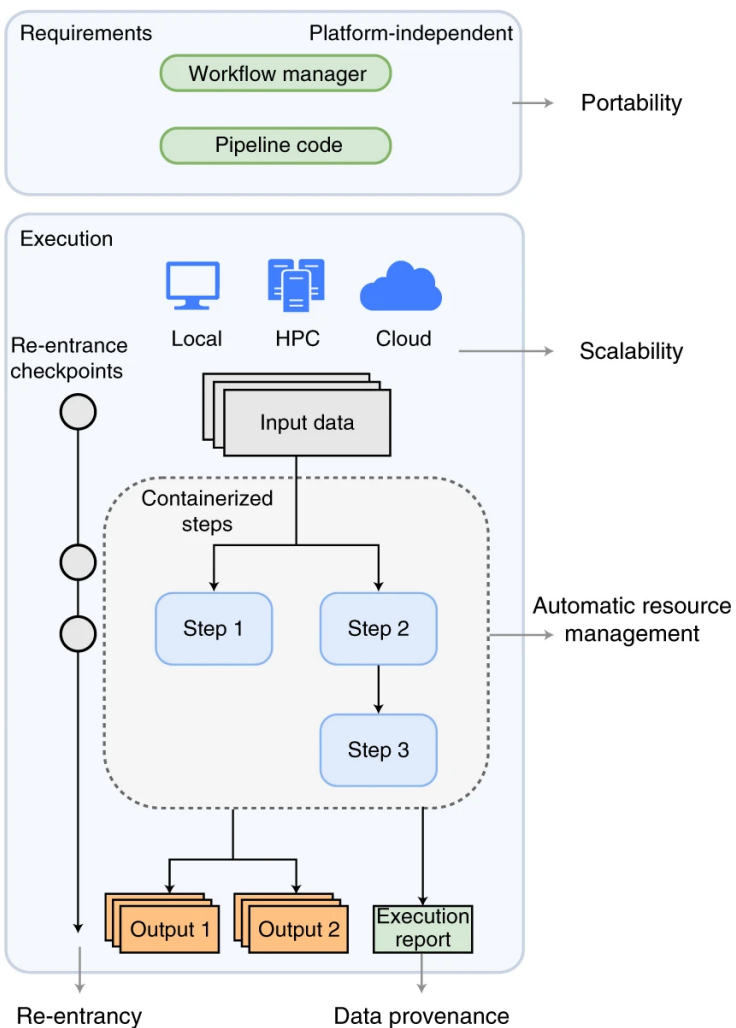
a Analysis workflow



b Traditional pipeline



c Workflow manager



Input data

Output data

Software, versions, parameters

! Fixed version, local compute environment

