

# Running Docker containers

INTRODUCTION TO DOCKER



**Tim Sangster**

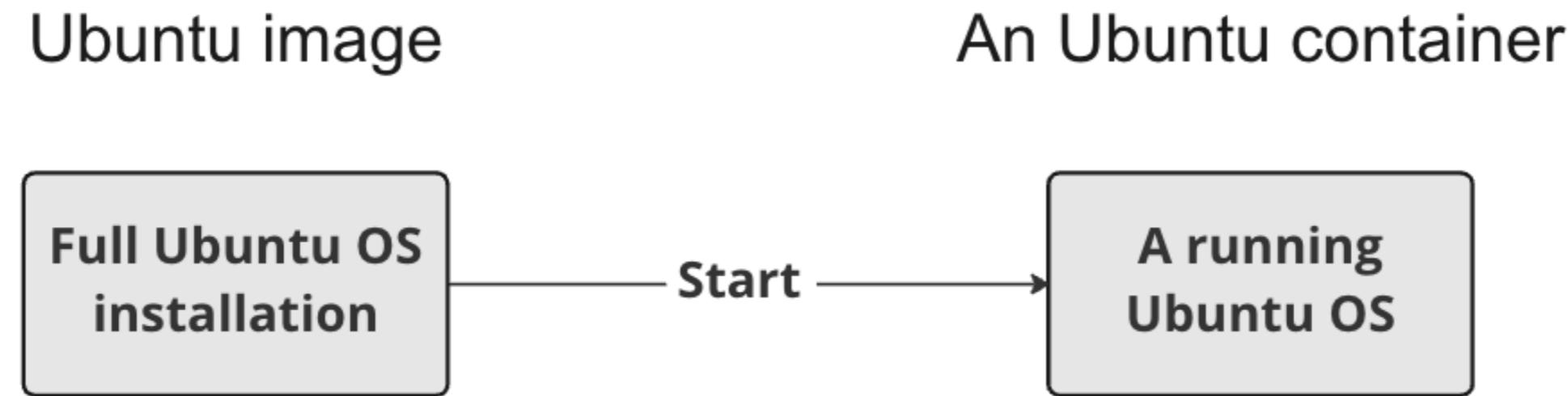
Software Engineer @ DataCamp

# Prerequisite

Command	Usage
<code>nano &lt;file-name&gt;</code>	Opens <code>&lt;file-name&gt;</code> in the nano text editor
<code>touch &lt;file-name&gt;</code>	Creates an empty file with the specified name
<code>echo "&lt;text&gt;"</code>	Prints <code>&lt;text&gt;</code> to the console
<code>&lt;command&gt; &gt;&gt; &lt;file&gt;</code>	Pushes the output of <code>&lt;command&gt;</code> to the end of <code>&lt;file&gt;</code>
<code>&lt;command&gt; -y</code>	Automatically respond yes to all prompts from <code>&lt;command&gt;</code>

# The Docker CLI

- Docker command line interface will send instructions to the Docker daemon.
- Every command starts with `docker`.



# Docker container output

```
docker run <image-name>
```

```
docker run hello-world
```

Hello from Docker!

To generate this message, Docker took the following steps:

1. The Docker client contacted the Docker daemon.
2. The Docker daemon created a new container from the hello-world image which runs the executable that produces the output you are currently reading.
3. The Docker daemon streamed that output to the Docker client, which sent it to your terminal.

# Choosing Docker container output

```
docker run <image-name>
```

```
docker run ubuntu
```

```
repl@host:/# docker run ubuntu
```

```
repl@host:/#
```

# An interactive Docker container

Adding `-it` to `docker run` will give us an interactive shell in the started container.

```
docker run -it <image-name>
```

```
docker run -it ubuntu
```

```
docker run -it ubuntu
repl@container:/#
```

```
repl@container:/# exit
exit
repl@host:/#
```

# Running a container detached

Adding `-d` to `docker run` will run the container in the background, giving us back control of the shell.

```
docker run -d <image-name>  
docker run -d postgres
```

```
repl@host:/# docker run -d postgres  
4957362b5fb7019b56470a99f52218e698b85775af31da01958bab198a32b072  
repl@host:/#
```

# Listing and stopping running containers

```
docker ps
```

```
repl@host:/# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
4957362b5fb7        postgres            "docker-entrypoint.s..."   About a minute ago
Up  About a minute   5432/tcp          awesome_curiel
```

```
docker stop <container-id>
```

```
repl@host:/# docker stop cf91547fd657
cf91547fd657
```

# Summary of new commands

Usage	Command
Start a container	<code>docker run &lt;image-name&gt;</code>
Start an interactive container	<code>docker run -it &lt;image-name&gt;</code>
Start a detached container	<code>docker run -d &lt;image-name&gt;</code>
List running containers	<code>docker ps</code>
Stop a container	<code>docker stop &lt;container-id&gt;</code>

# **Let's practice!**

**INTRODUCTION TO DOCKER**

# Working with Docker containers

INTRODUCTION TO DOCKER



**Tim Sangster**

Software Engineer @ DataCamp

# Listing containers

```
repl@host:/# docker ps
```

CONTAINER ID	IMAGE	CREATED	STATUS	NAMES
3b87ec116cb6	postgres	2 seconds ago	Up 1 second	adoring_germain
8a7830bbc787	postgres	3 seconds ago	Up 2 seconds	exciting_heisenberg
fefdf1687b39	postgres	3 seconds ago	Up 2 seconds	vigilant_swanson
b70d549d4611	postgres	4 seconds ago	Up 3 seconds	nostalgic_matsumoto
a66c71c54b92	postgres	4 seconds ago	Up 4 seconds	lucid_matsumoto
8d4f412adc3f	postgres	6 seconds ago	Up 5 seconds	fervent_ramanujan
fd0b3b2a843e	postgres	7 seconds ago	Up 6 seconds	cool_dijkstra
0d1951db81c4	postgres	8 seconds ago	Up 7 seconds	happy_sammet
...				

# Named containers

```
docker run --name <container-name> <image-name>
```

```
repl@host:/# docker run --name db_pipeline_v1 postgres
repl@host:/# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
43aa37614330        postgres            "docker-entrypoint.s..."   About a minute ago
Up  About a minute          5432/tcp          db_pipeline_v1
STATUS                NAMES
```

```
docker stop <container-name>
```

```
repl@host:/# docker stop db_pipeline_v1
```

# Filtering running containers

```
docker ps -f "name=<container-name>"
```

```
repl@host:/# docker ps -f "name=db_pipeline_v1"
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
43aa37614330        postgres            "docker-entrypoint.s..."   About a minute ago
Up  About a minute   5432/tcp          db_pipeline_v1
```

# Container logs

```
docker logs <container-id>
```

```
repl@host:/# docker logs 43aa37614330
```

```
The files belonging to this database system will be owned by user "postgres".
```

```
This user must also own the server process.
```

```
The database cluster will be initialized with locale "en_US.utf8".
```

```
The default database encoding has accordingly been set to "UTF8".
```

```
PostgreSQL init process complete; ready for start up.
```

```
2022-10-24 12:10:40.318 UTC [1] LOG:  database system is ready to accept connect..
```

# Live logs

```
docker logs -f <container-id>
```

```
repl@host:/# docker logs -f 43aa37614330
PostgreSQL init process complete; ready for start up.
```

```
2022-10-24 12:10:40.309 UTC [1] LOG:  starting PostgreSQL 14.5 (Debian 14.5-1.pg..
2022-10-24 12:10:40.309 UTC [1] LOG:  listening on IPv4 address "0.0.0.0", port ..
2022-10-24 12:10:40.309 UTC [1] LOG:  listening on IPv6 address ":::", port 5432
2022-10-24 12:10:40.311 UTC [1] LOG:  listening on Unix socket "/var/run/postgre..
2022-10-24 12:10:40.315 UTC [62] LOG:  database system was shut down at 2022-10-..
2022-10-24 12:10:40.318 UTC [1] LOG:  database system is ready to accept connect..
```

# Cleaning up

```
docker container rm <container-id>
```

```
repl@host:# docker stop 43aa37614330  
43aa37614330  
repl@host:# docker container rm 43aa37614330  
43aa37614330
```

# Summary of new commands

Usage	Command
Start container with a name	<code>docker run --name &lt;container-name&gt; &lt;image-name&gt;</code>
Filter running container on name	<code>docker ps -f "name=&lt;container-name&gt;"</code>
See existing logs for container	<code>docker logs &lt;container-id&gt;</code>
See live logs for container	<code>docker logs -f &lt;container-id&gt;</code>
Exit live log view of container	<code>CTRL+C</code>
Remove stopped container	<code>docker container rm &lt;container-id&gt;</code>

# **Let's practice!**

**INTRODUCTION TO DOCKER**

# Managing local docker images

INTRODUCTION TO DOCKER

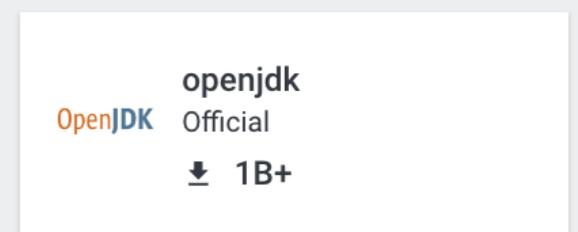
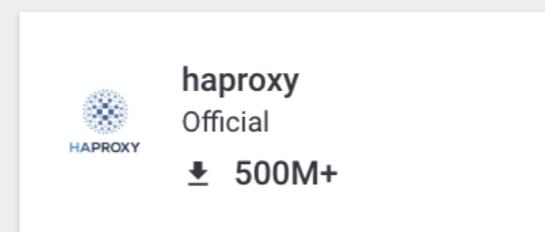
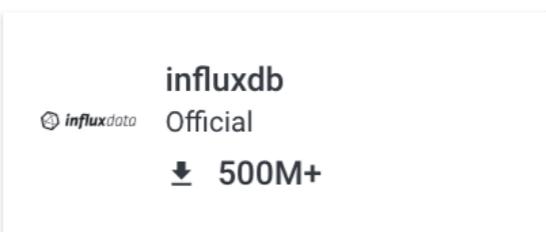
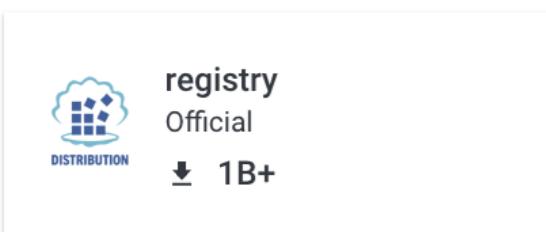
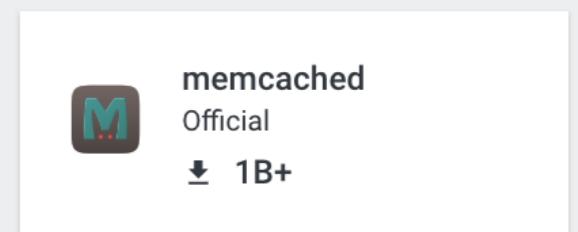
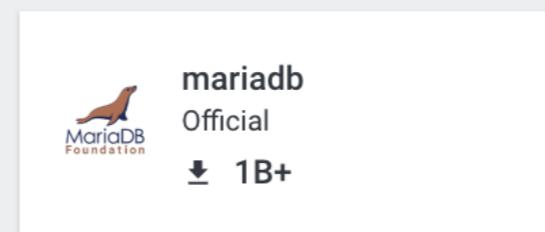
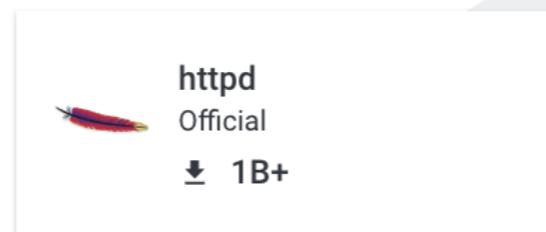
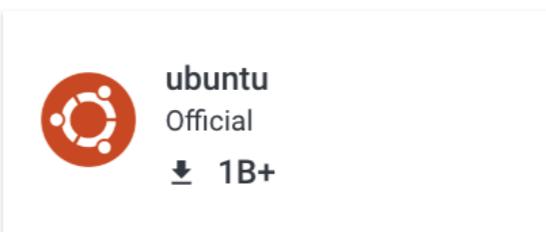
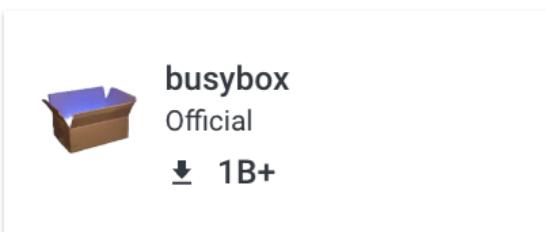


**Tim Sangster**

Software Engineer @ DataCamp

# Docker Hub is the world's largest library and community for container images

Browse over 100,000 container images from software vendors, open-source projects, and the community.



# Pulling an image

```
docker pull <image-name>
```

```
docker pull postgres  
docker pull ubuntu
```

```
repl@host:/# docker pull hello-world  
Using default tag: latest  
latest: Pulling from library/hello-world  
7050e35b49f5: Pull complete  
Digest: sha256:e18f0a777aefabe047a671ab3ec3eed05414477c951ab1a6f352a06974245fe7  
Status: Downloaded newer image for hello-world:latest  
docker.io/library/hello-world:latest
```

# Image versions

Supported tags and respective [Dockerfile](#) links

- [18.04](#), [bionic-20221019](#), [bionic](#)
- [20.04](#), [focal-20221019](#), [focal](#)
- [22.04](#), [jammy-20221020](#), [jammy](#), [latest](#)
- [22.10](#), [kinetic-20221024](#), [kinetic](#), [rolling](#)
- [14.04](#), [trusty-20191217](#), [trusty](#)
- [16.04](#), [xenial-20210804](#), [xenial](#)

```
docker pull <image-name>:<image-version>
```

```
docker pull ubuntu:22.04
```

```
docker pull ubuntu:jammy
```

# Listing images

docker images

```
repl@host:/# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
hello-world	latest	46331d942d63	7 months ago	9.14kB
ubuntu	bionic-20210723	7c0c6ae0b575	15 months ago	56.6MB
postgres	12.7	f076c2fa35f5	15 months ago	300MB
postgres	10.3	cbb7481ff9d5	4 years ago	232MB
...				

# Removing images

```
docker image rm <image-name>
```

```
repl@host:/# docker image rm hello-world
Untagged: hello-world:latest
Untagged: hello-world@sha256:e18f0a777aefabe047a671ab3ec3eed05414477c951ab1a6f35..
Deleted: sha256:46331d942d6350436f64e614d75725f6de3bb5c63e266e236e04389820a234c4
Deleted: sha256:efb53921da3394806160641b72a2cbd34ca1a9a8345ac670a85a04ad3d0e3507
```

```
repl@host:/# docker image rm hello-world
Error response from daemon: conflict: unable to remove repository reference
"hello-world" (must force) - container 96a7b7b0c535 is using its
referenced image 46331d942d63
```

# Cleaning up containers

docker container prune

```
repl@host:/# docker container prune
WARNING! This will remove all stopped containers.
Are you sure you want to continue? [y/N] y
Deleted Containers:
4a7f7eebae0f63178aff7eb0aa39cd3f0627a203ab2df258c1a00b456cf20063
f98f9c2aa1leaf727e4ec9c0283bc7d4aa4762fdbba7f26191f26c97f64090360
Total reclaimed space: 212 B
```

# Cleaning up images

```
docker image prune -a
```

```
repl@host:/# docker image prune -a
WARNING! This will remove all images without at least one container associated t..
Are you sure you want to continue? [y/N] y
Deleted Images:
untagged: alpine:latest
untagged: alpine@sha256:3dcdb92d7432d56604d4545cbd324b14e647b313626d99b889d0626d..
deleted: sha256:4e38e38c8ce0b8d9041a9c4fefef786631d1416225e13b0bfe8cfa2321aec4bba
deleted: sha256:4fe15f8d0ae69e169824f25f1d4da3015a48feeeeabb265cd2e328e15c6a869f

Total reclaimed space: 16.43 MB
```

# Dangling images

docker images

```
repl@host:/# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
testsql	latest	6c49f0cce145	7 months ago	3.73GB
<none>	<none>	a22b8450b88f	7 months ago	3.73GB
<none>	<none>	10dd2d03f59c	7 months ago	3.73GB
<none>	<none>	878bae40320b	7 months ago	3.73GB
<none>	<none>	4ea70583ba54	7 months ago	3.75GB
<none>	<none>	3c64576a3a7d	7 months ago	3.75GB

# Summary of new commands

Usage	Command
Pull an image	<code>docker pull &lt;image-name&gt;</code>
Pull a specific version of an image	<code>docker pull &lt;image-name&gt;:&lt;image-version&gt;</code>
List all local images	<code>docker images</code>
Remove an image	<code>docker image rm &lt;image-name&gt;</code>
Remove all stopped containers	<code>docker container prune</code>
Remove all images	<code>docker image prune -a</code>

# **Let's practice!**

**INTRODUCTION TO DOCKER**