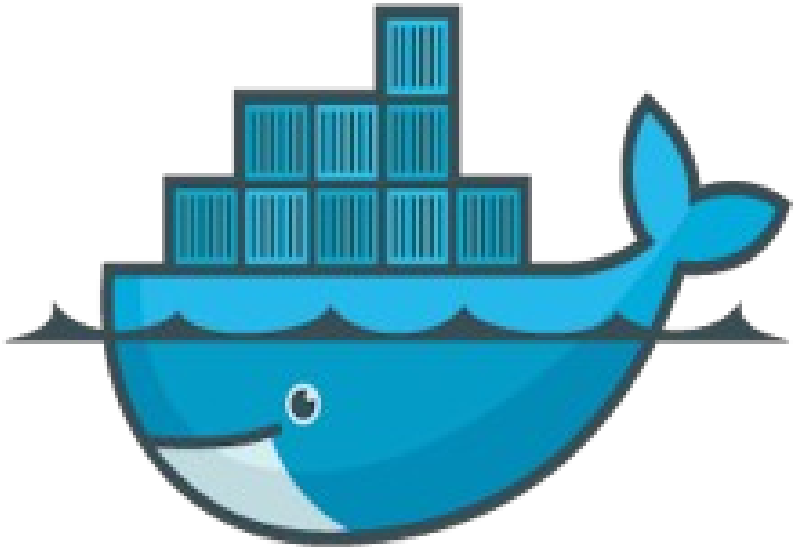
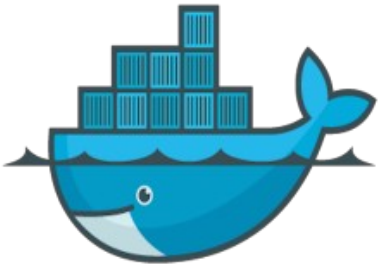


Wat is Docker?

Wat is Docker?

Goedenavond, ik ben Ron Regeer.
Kernlid van de Linux GG van Zevenaar.



Vanavond wil ik jullie het een en ander vertellen over:

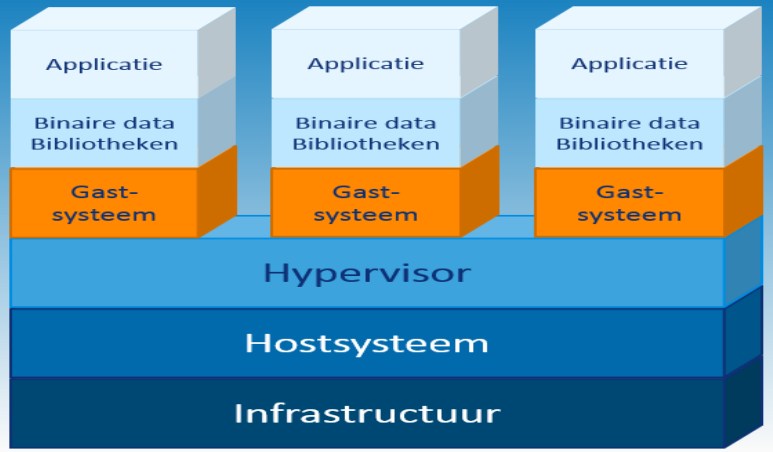
- Docker
- Images
- Containers
- Docker Hub
- Container starten
- Zelf een image maken
- Docker commando’s
- Beheer met Portainer

Deze presentatie is straks te downloaden als pdf.

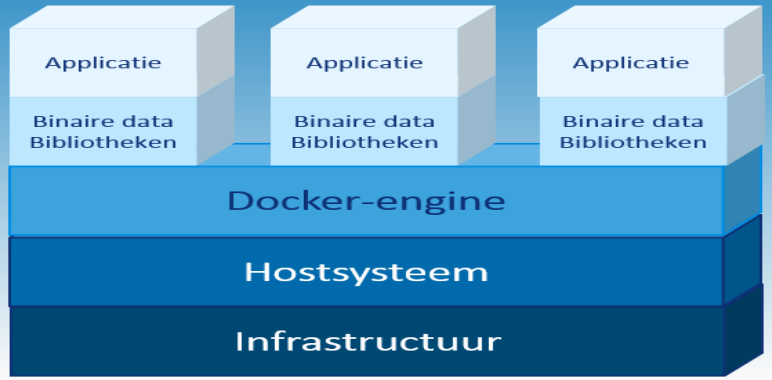
Containers

Docker is kort gezegd een open-source raamwerk waarmee het mogelijk wordt een applicatie in een lichtgewicht, verplaatsbare (software)container te verpakken. Zo wordt het installeren van een applicatie op een server even eenvoudig als het installeren van een mobiele app op je tablet of smartphone….

Docker een ~~virtual machine~~ engine



Virtuele Machines (VM's)



Docker-container

Docker een ~~virtual machine~~ engine

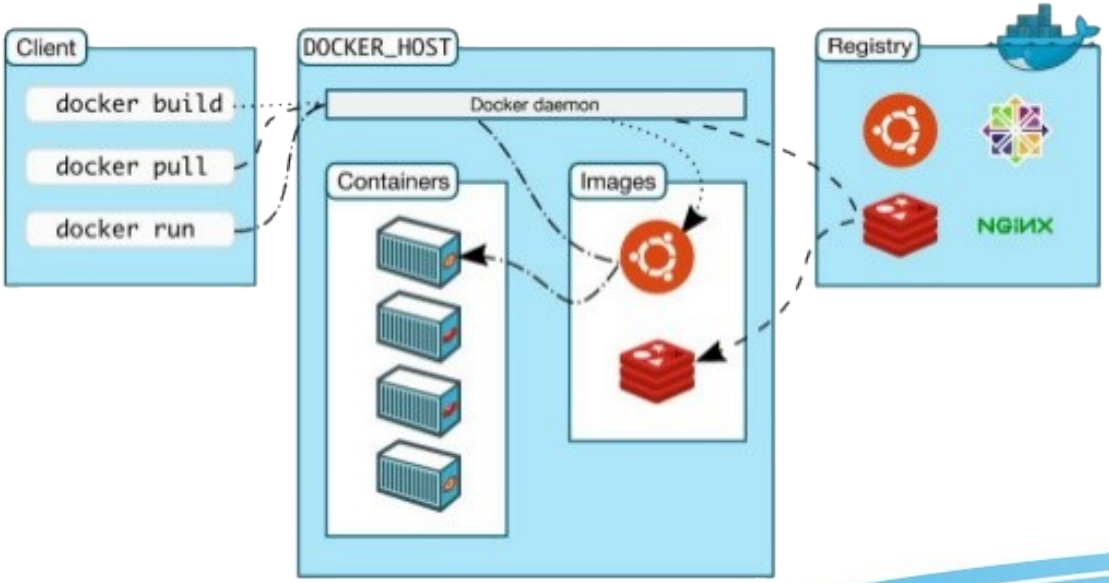
Een **V**irtual **M**achine kan je zien als een compleet huis. Het heeft eigen voorzieningen zoals verwarming, sanitaire installatie, airco, gas en licht aansluiting.. Het functioneert als één geheel omdat het alle benodigde onderdelen zelf heeft.

Docker een ~~virtual machine~~ engine

Een Docker container kan je zien als een appartement in een flatgebouw. Docker Engine regelt de toegang. Die laat de bewoners in en uit het gebouw. De appartementen (containers) delen de resources van het flatgebouw zoals de verwarming en de diverse leidingen en afvoeren in het gebouw.

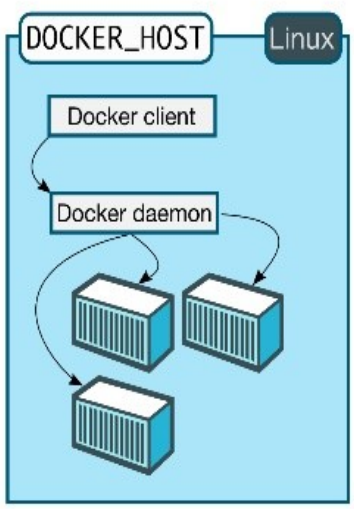
Docker is dus GEEN VM !!

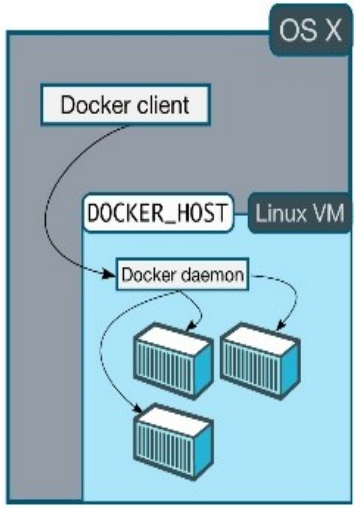
Docker Components

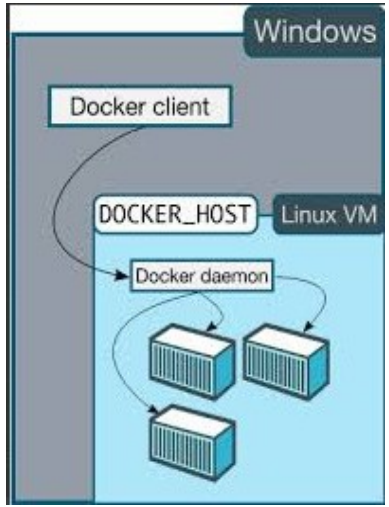


Docker draait op Linux,Windows en Mac

Docker draait native onder Linux en maakt gebruik van verschillende functies van de Linux kernel om zijn functionaliteit te leveren.
Wil je Docker onder Windows of OS X gebruiken dan wordt dat via een Linux VM gedaan
De Linux VM zorgt voor de kernel en de speciale functies die alleen in Linux aanwezig zijn.

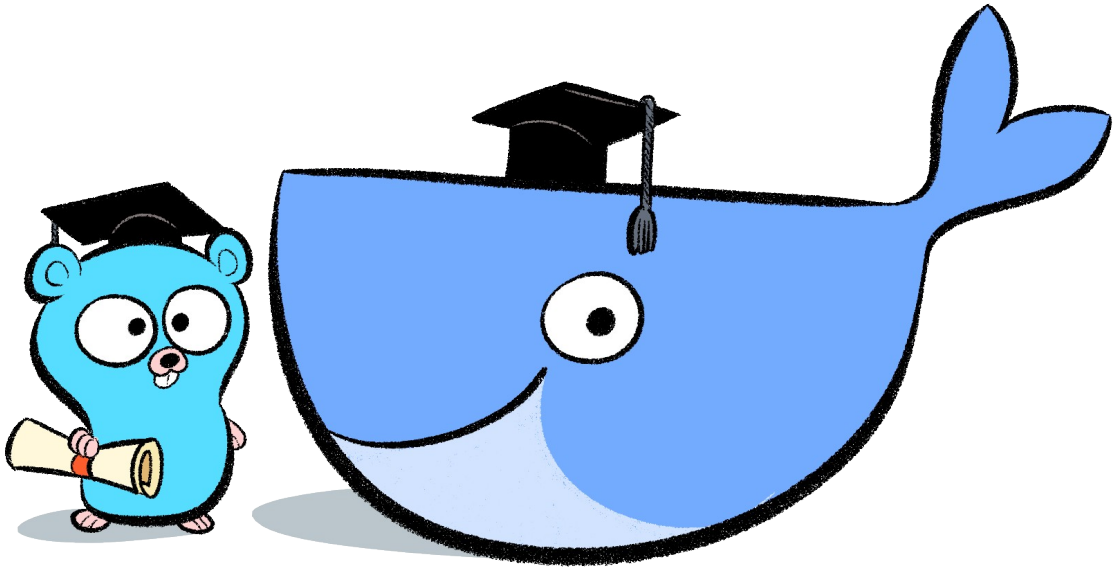






Wat zijn die speciale Linux functies?

- Namespaces
- Control groups
- Union file systems



Namespaces

Deze namespaces bieden een isolatielaag. Elk aspect van een container wordt in een afzonderlijke namespace uitgevoerd en de toegang is beperkt tot die namespace.

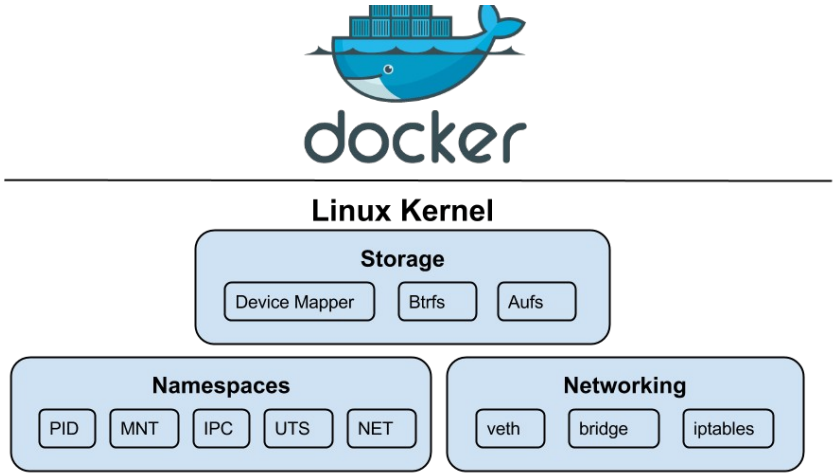
pid : Process isolation (PID: Process ID).

net: Managing network interfaces (NET: Networking).

ipc: Managing access to IPC resources (InterProcessCommunication).

mnt: Managing filesystem mount points (MNT: Mount).

uts: Isolating kernel and version identifiers. (Unix Timesharing System).

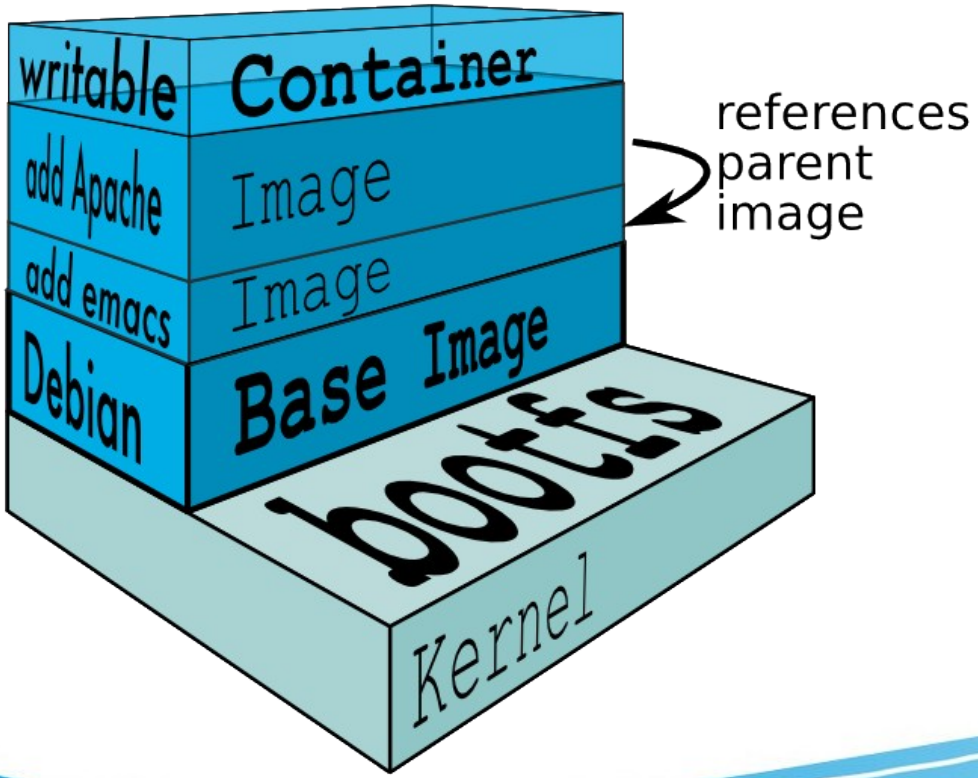


Control group

Een control group beperkt een toepassing tot een specifieke reeks bronnen.

Je kunt bijvoorbeeld het beschikbare geheugen beperken tot een specifieke container.

Union File Systems

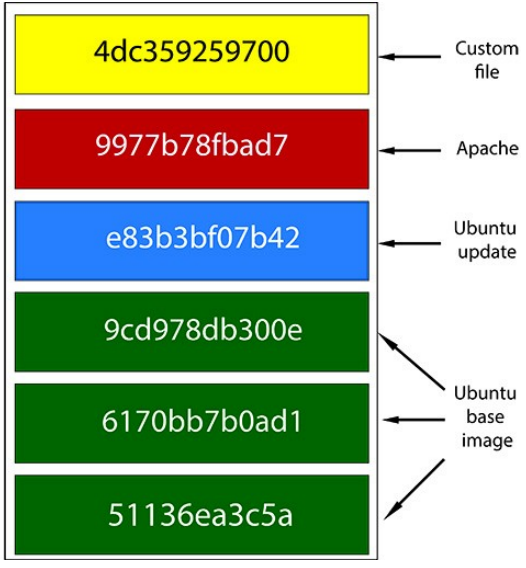


Union-bestandssystemen zijn bestands-systemen die werken door lagen te maken.

Alleen de bovenste laag is beschrijfbaar

Docker image

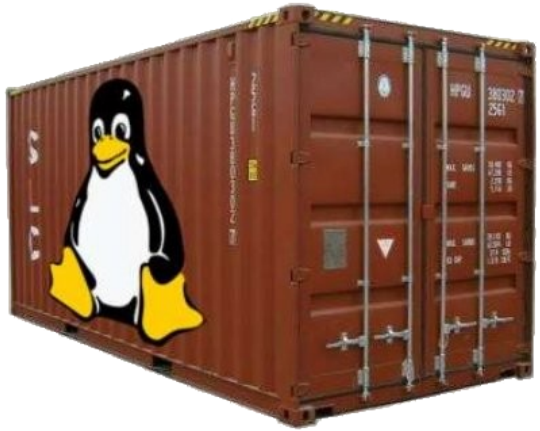
Een Docker-image bevat meerdere lagen. Alleen de bovenste laag is beschrijfbaar.
De Docker-images worden laag voor laag opgebouwd. Als de laag al is gedownload, wordt deze opnieuw gebruikt.
Het gebruik van een tag wordt aanbevolen, zoals b.v. ubuntu: 17.04.




```
$docker pull ubuntu:17.04
17.04: Pulling from library/ubuntu
c2ca09a1934b: Pull complete
d6c3619d2153: Pull complete
0efe07335a04: Pull complete
6b1bb01b3a3b: Pull complete
43a98c187399: Pull complete
Digest:
sha256:5d41c289942008211c2964bca72800f5c9d5ea5aa4057528da617fb36463d4a
b
Status: Downloaded newer image for ubuntu:17.04
```

Container

Docker Engine combineert de namespaces, control groups, en UnionFS. Het resultaat is wat we een container noemen.



Samengevat

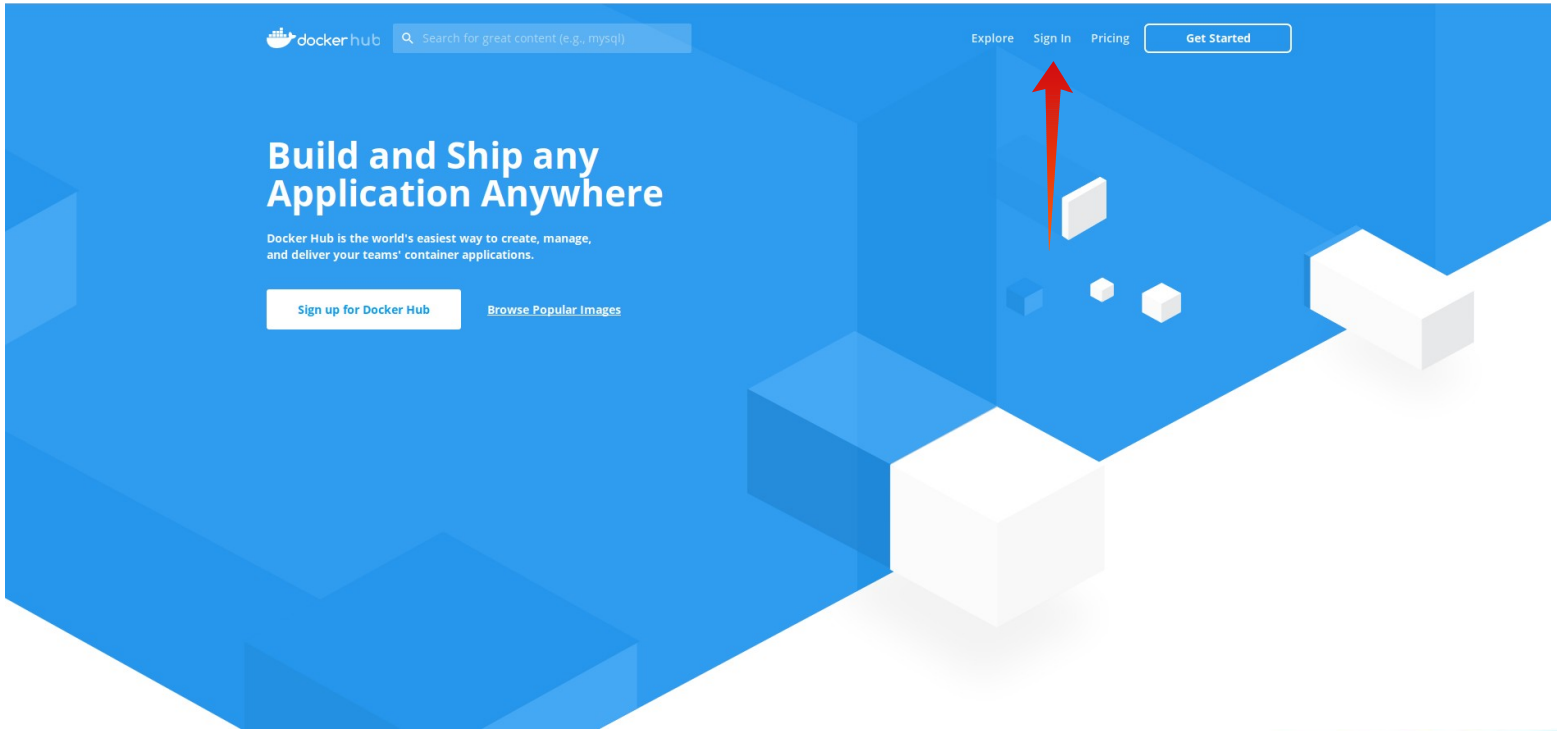
De container-technologie is niet alleen een efficiënt alternatief voor de klassieke hardwarevirtualisatie. Een container die op Ubuntu is gemaakt draait net zo makkelijk op Centos of Arch of wat voor Linux versie dan ook.
Het schalen van containers is ook een stuk makkelijker omdat het opstarten en verwijderen heel snel gaat.
Ook de systeembelasting van containers is lager dan van een complete VM.
Resultaat is dat er meer software kan draaien op dezelfde hardware. Dat is dus efficiënter en kostenbesparend!

Docker Hub

Docker Hub is een repository waar je images vindt die kunnen worden gebruikt om containers te maken. Er zijn officiële en openbare repositories.

Wij gaan straks zelf een eigen repository op de Docker Hub maken.

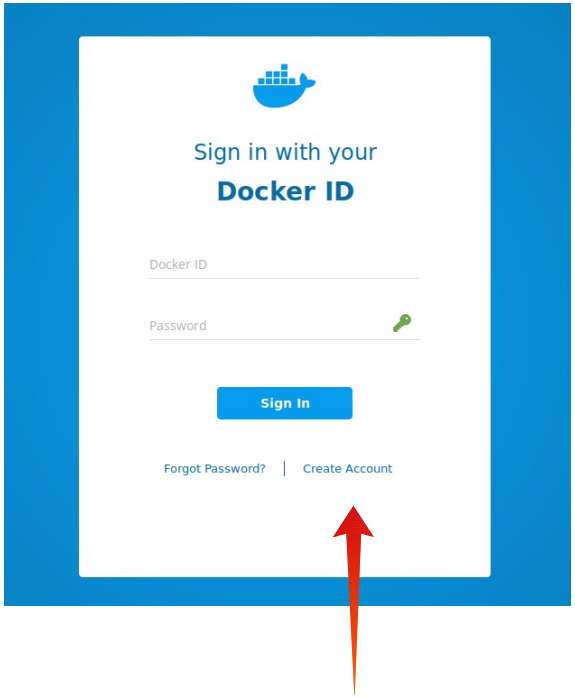
Docker Hub



https://hub.docker.com

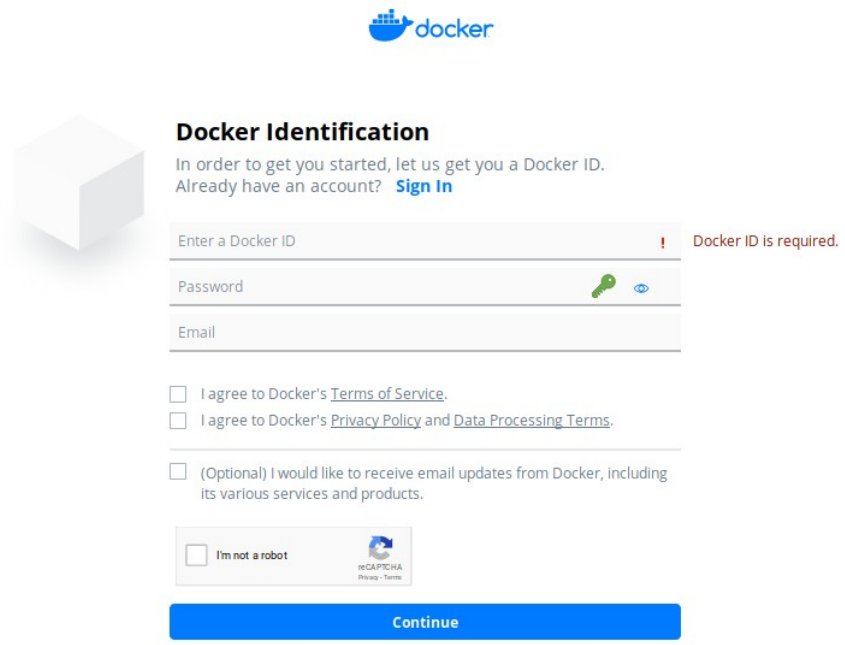
Via deze url kan je een account aanmaken en ook inloggen op je account. (klik Sign In)
Een account is gratis.
Je kan ook een betaald account nemen.

Docker Hub



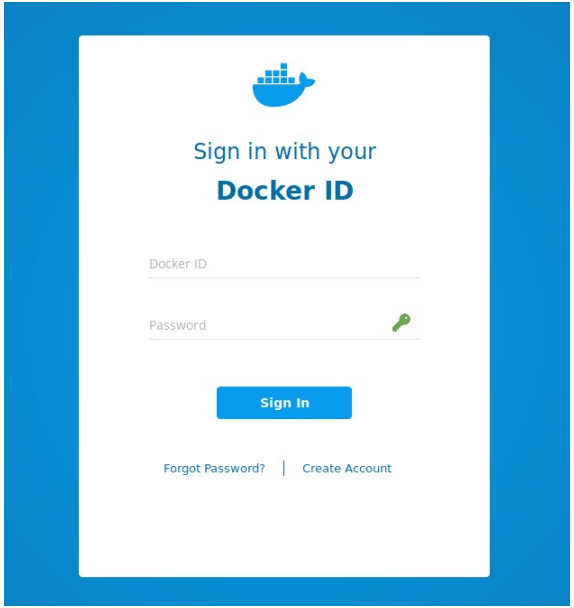
Omdat we nog geen Docker ID hebben klikken we op Create Account

Docker Hub

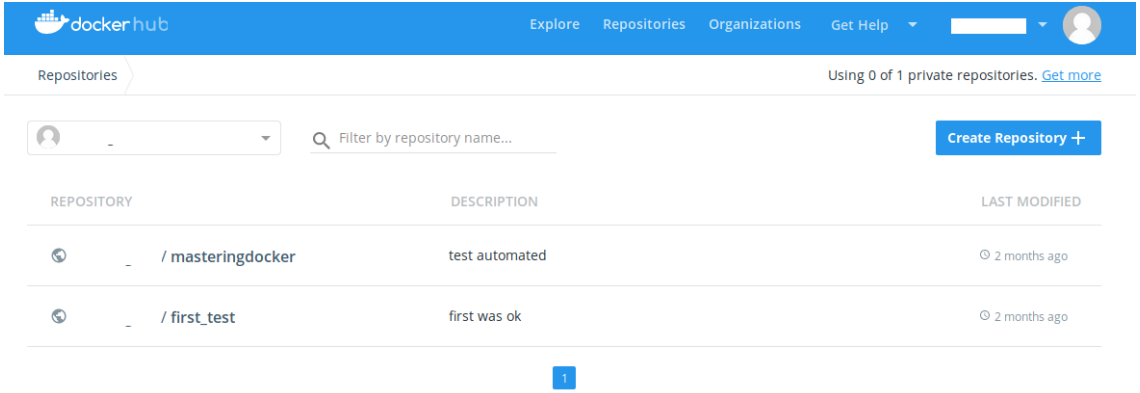


Hier vullen we de gevraagde gegevens in. Docker ID moet een unieke naam zijn. Geef een sterk wachtwoord op en je emailadres. Vink de rest aan. Klik op Continue en de rest wijst zich vanzelf. Gefeliciteerd! Je hebt een Docker ID.

Docker Hub



Nu kunnen we inloggen met ons Docker ID en wachtwoord op de Docker Hub.



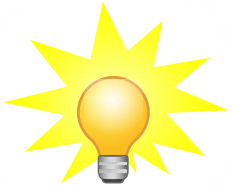
Docker engine installeren

Maar om verder te kunnen werken dienen we ook Docker op onze computer te installeren.

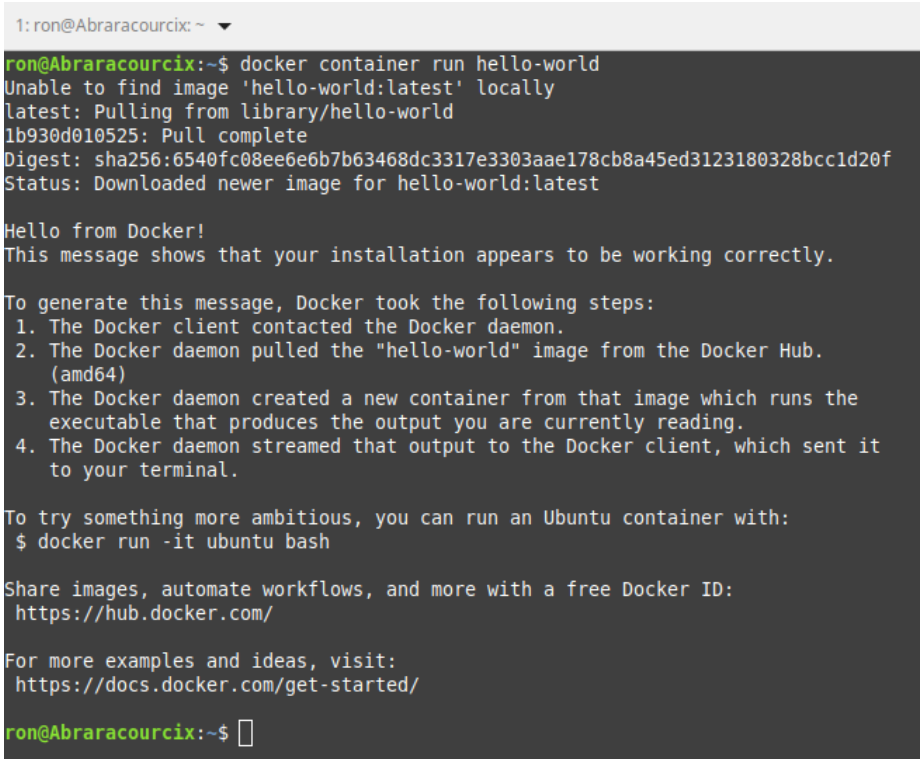
Installeren kan via:

- Repository van Docker ,zie de docker website.
- De repository van je eigen distro.
- Via snaps: snap install docker

Als Docker is geïnstalleerd kunnen we via de terminal allerlei commando's geven. Dit moeten we altijd doen met sudo <commando>. Als je dit vervelend vindt dan geeft de volgende sheet de oplossing.



Docker runnen



Start een terminal en geef het commando: sudo usermod -aG docker $USER Vul voor $USER je eigen gebruikersnaam in. Nu hoef je voor elk docker commando geen sudo meer te typen... Dan even testen of het werkt.. type commando:
docker container run hello-world
Docker vindt geen images en zoekt ze dan op de docker hub. Doet een pull naar je eigen computer en runt dan de container.
Door deze korte test weet je dat Docker werkt.

Docker runnen

We gaan nu als voorbeeld een webserver op onze computer installeren. Dit doen we met het commando docker run --rm prakhar1989/static-site. *(Dit is zomaar een voorbeeld)* We zien dat de docker client via een pull de benodigde bestanden ophaalt en dan de container start. Op dit moment is de terminal bezet. Sluiten we de terminal dan stopt de container ook.

```
ron@Abraracourcix:~$ docker run --rm prakhar1989/static-site
Unable to find image 'prakhar1989/static-site:latest' locally
latest: Pulling from prakhar1989/static-site
[DEPRECATION NOTICE] registry v2 schema1 support will be removed in an upcoming rel
ruption.
d4bce7fd68df: Pull complete
a3ed95caeb02: Pull complete
573113c4751a: Pull complete
31917632be33: Pull complete
77e66f18af1c: Pull complete
df3f108f3ade: Pull complete
d7a279eb19f5: Pull complete
e798589c05c5: Pull complete
78eeaf458ae0: Pull complete
Digest: sha256:48ca6254c16b81a7960ad874e901f027fbcaac66509cfd79ce8f3f6da424d3b1
Status: Downloaded newer image for prakhar1989/static-site:latest
Nginx is running...
```

Docker runnen

```
ron@Abraracourcix:~$ docker ps
CONTAINER ID        IMAGE                      COMMAND             CREATED             STATUS              PORTS               NAMES
7d84b431199c        prakhar1989/static-site    "./wrapper.sh"      17 seconds ago      Up 17 seconds       80/tcp, 443/tcp     beautiful_meninsky
ron@Abraracourcix:~$
```

Ik heb een andere terminal geopend en het commando docker ps gegeven. Je ziet dat de container runt en dat de poorten 80 en 443 aanwezig zijn. Ze worden echter nog niet doorgegeven aan ons systeem zodat we ze op dit moment niet kunnen bekijken.
Type CTRL+C om de container te stoppen.

We gaan nu zorgen dat we de poorten wel doorgeven. We starten opnieuw met commando:
docker run -d -P --name static-site prakhar1989/static-site
-d start de container in detached mode (je kan de terminal daarna gewoon sluiten)
-P zet de poorten die in de image zijn gedefinieerd door naar een willekeurige poort op het systeem.

Docker runnen

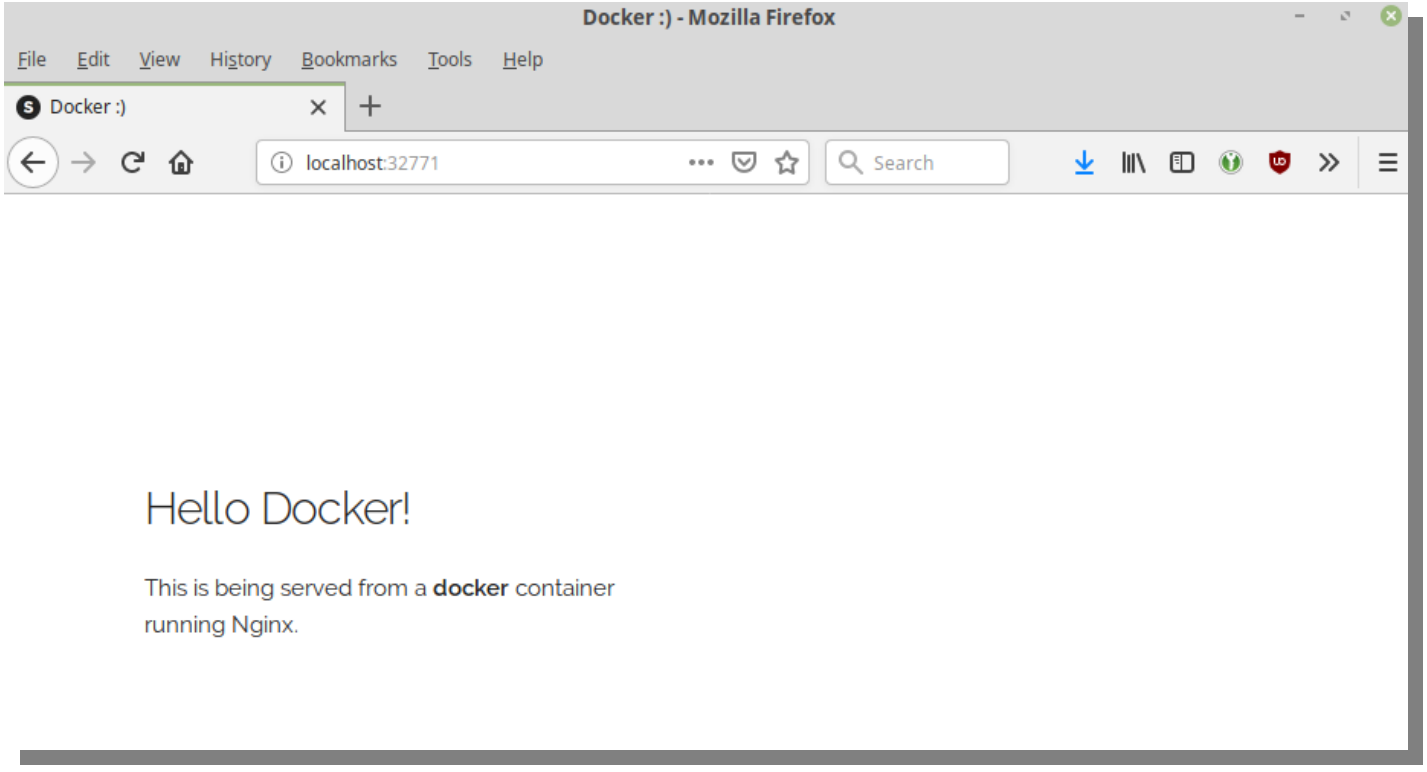
```
ron@Abraracourcix:~$ docker run -d -P --name static-site prakhar1989/static-site
15d247ad3c763e514e9e885835125c53286bc512399c0feb53a3e6795d3e9c9a
ron@Abraracourcix:~$
```

```
ron@Abraracourcix:~$ docker ps
CONTAINER ID        IMAGE                  COMMAND             CREATED             STATUS              PORTS                                          NAMES
bd0779a4ea98        prakhar1989/static-site   "./wrapper.sh"      14 seconds ago      Up 12 seconds       0.0.0.0:32771->80/tcp, 0.0.0.0:32770->443/tcp   static-site
ron@Abraracourcix:~$
```

Bij ports zien we dat poort 80 gekoppeld is met 32771 op het host systeem. Als we dit in onze browser ingeven dan kunnen we zien wat de webserver op dit moment doet.

Docker runnen

We starten onze browser en typen deze regel in: http://localhost:32771



Het werkt!
Dit korte voorbeeld laat zien dat we op een vrij eenvoudige manier een webserver op ons systeem kunnen zetten!

Het is dus wel leuk om hier mee te experimenteren.

Onze eerste image

In het volgende voorbeeld gaan we zelf een image maken met het build commando.
Open een terminal. Type de volgende commando's
mkdir static_web
cd static_web
vim Dockerfile *(Gebruik de editor die je zelf gewend bent)*
Type in je editor de volgende regels.

```
# Version: 0.0.1
FROM ubuntu:16.04
LABEL maintainer="Vul hier je eigen naam in"
RUN apt-get update; apt-get install -y nginx
RUN echo 'Hi, I am in your container' \
 >/var/www/html/index.html
EXPOSE 80
```

Save je invoer en verlaat de editor.

Onze eerste image

Uitleg van de regels in de Dockerfile:

	Uitleg:
`# Version: 0.0.1`	# is een comment regel
`FROM ubuntu:16.04`	1e commando MOET altijd een FROM zijn
`LABEL maintainer="eigen naam/email"`	Dit is niet verplicht
`RUN apt-get update; apt-get install -y nginx`	Update Ubuntu en installeert nginx
`RUN echo 'Hi, I am in your container' \ >/var/www/html/index.html`	Zet deze tekst in index.html
`EXPOSE 80`	Gebruik poort 80 van container

Nu gaan we een image maken met het commando build:

`docker build -t="jedockerid/static_web" .` *(je docker id is bv je eigen naam)*

Vergeet de punt niet!

Onze eerste image

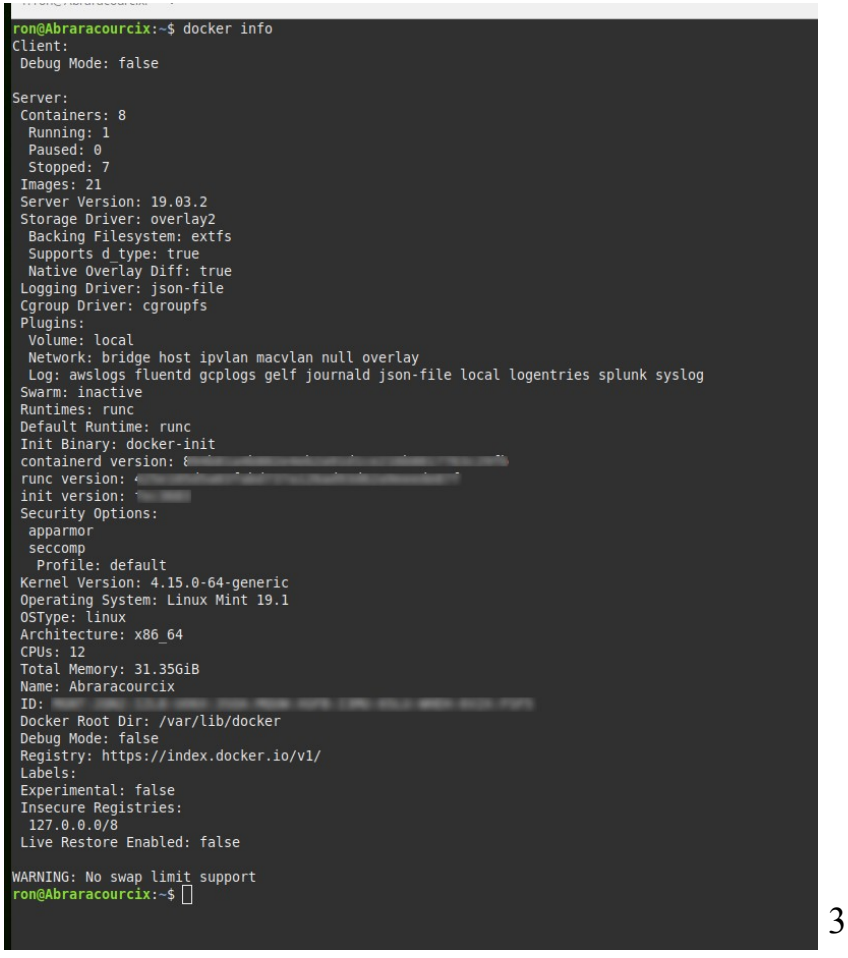
docker run -d -p 80 --name static_web jedockerid/static_web nginx -g "daemon off;"
Eerst even kijken welke poort je moet opgeven. Commando docker ps

Docker commando’s

docker info

Met deze opdracht wordt systeembrede informatie over de Docker-installatie weergegeven. De weergegeven informatie omvat o.a. de kernelversie, het aantal containers en images. Het aantal getoonde images is het aantal unieke images. Dus dezelfde image met een andere tag wordt slechts één keer geteld.

I



Docker commando’s

Images:

docker image pull <image>	pull een image
docker image ls	overzicht van images
docker image rm <image>	delete een image

docker image build --file <path_to_Dockerfile> --tag <REPOSITORY>:<TAG> .

<REPOSITORY> is normaal gesproken je dockerhub naam

Vergeet de punt aan het eind niet! Docker maakt image in de huidige directory.

Containers:

docker container ls	list running containers
docker container ls -a	list all containers
docker cointainer rm <container>	delete container
docker info	Check dat de docker binary werkt

Dit is slechts een (kleine) greep uit de commando’s.

Docker gegevens delen/bewaren

Om gegevens te delen of te bewaren kan een volume worden gebruikt.
Er zijn 3 manieren om een volume aan een Docker-container te koppelen:

Het volume van de container zelf

Hostmap gemount op de container

Gedeeld volume van andere container

```
$echo "Share me" > /tmp/share.txt

$docker run -v /tmp:/tmp bash -c "cat /tmp/share.txt"
share me

$docker run -v /tmp:/tmp bash -c "echo 'share me more' >> /tmp/share.txt"

$cat /tmp/share.txt
share me
share me more

$docker run -v /tmp:/data --name data-container bash

$docker run --volumes-from data-container bash -c "cat /data/share.txt"
share me
share me more
```

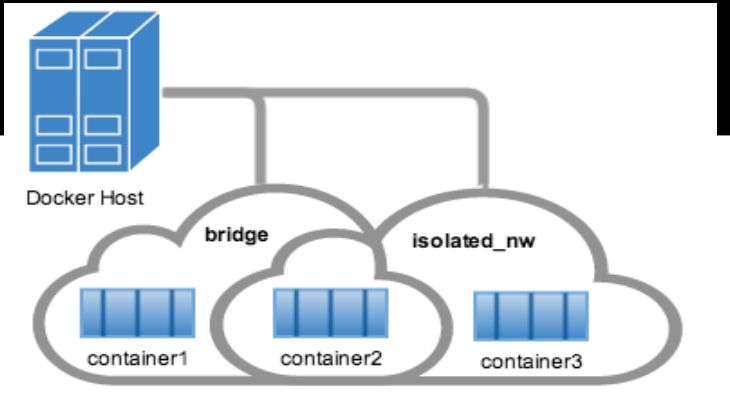
Docker Networking

Alle containers zijn standaard verbonden met bridge0 op de host.
De poorttoewijzing kan worden bepaald bij de start van de container.

```
$docker run -d -p 8080:80 nginx:1.11.8
$open http://localhost:8080

$docker run -d -P nginx:1.11.8
$docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
0b4cb1b3891a        nginx:1.11.8        "nginx -g 'daemon off"   7 seconds ago       Up 6 seconds        0.0.0.0:32769->80/tcp

$docker port 0b4cb1b3891a
443/tcp -> 0.0.0.0:32768
80/tcp -> 0.0.0.0:32769
```



Docker beheren met een webinterface

Het is niet altijd even handig om enkel maar via de cli te werken.
Er zijn diverse oplossingen bedacht om je Docker images en containers meer visueel te beheren.
Dit vindt plaats via een webinterface.
Er zijn diverse programma's die dit doen:

- Kitematic (voor Windows en Mac)
- Portainer
- Shipyard

In de volgende schermen gaan we Portainer wat verder bespreken.

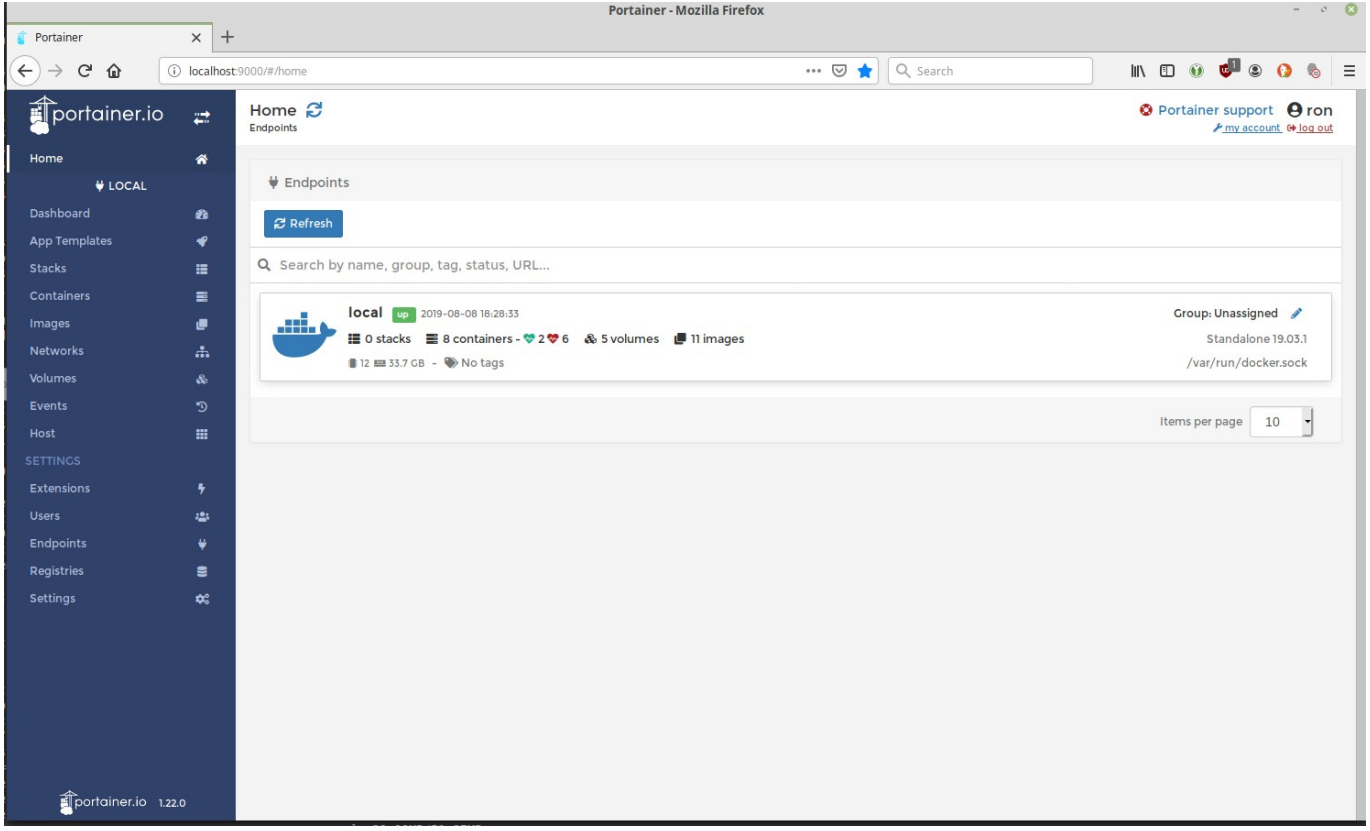
Docker beheren met Portainer

Portainer is een webbased hulpmiddel om via een grafische interface je containers te beheren.
Het is zelf een container, dus de installatie is vrij simpel.
Type in een terminal:
docker pull portainer/portainer
docker volume create portainer_data
docker run -d -p 9000:9000 --name portainer --restart always -v
/var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer

De interface is nu via je webbrowser te benaderen via poort 9000. Dus localhost:9000
Vul de inloggegevens in. En proberen maar.
Documentatie is te verkrijgen via https://portainer.readthedocs.io/en/latest/index.html
Deze container start automatisch als je inlogt op je machine.

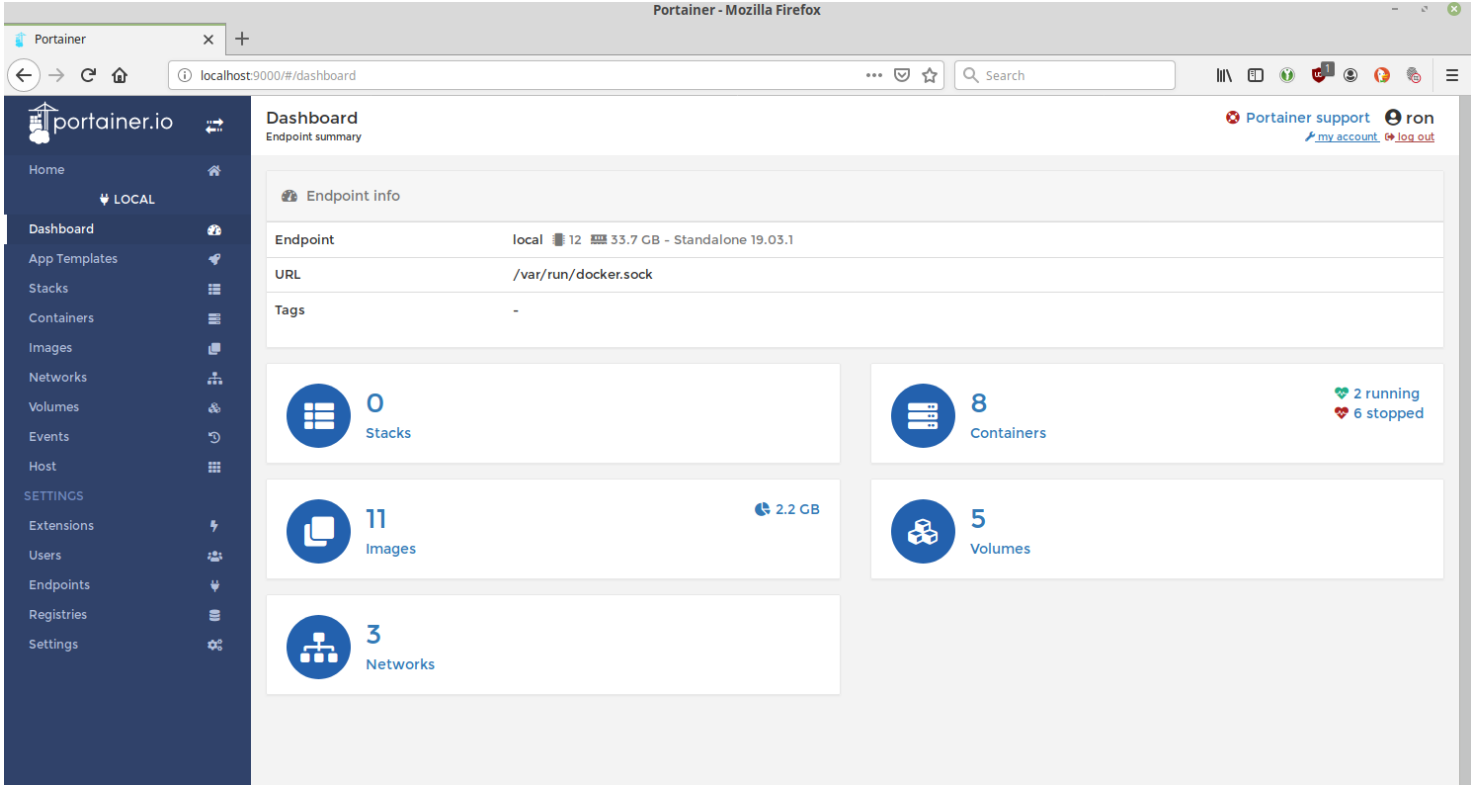
Docker beheren met Portainer

De homepage van Portainer



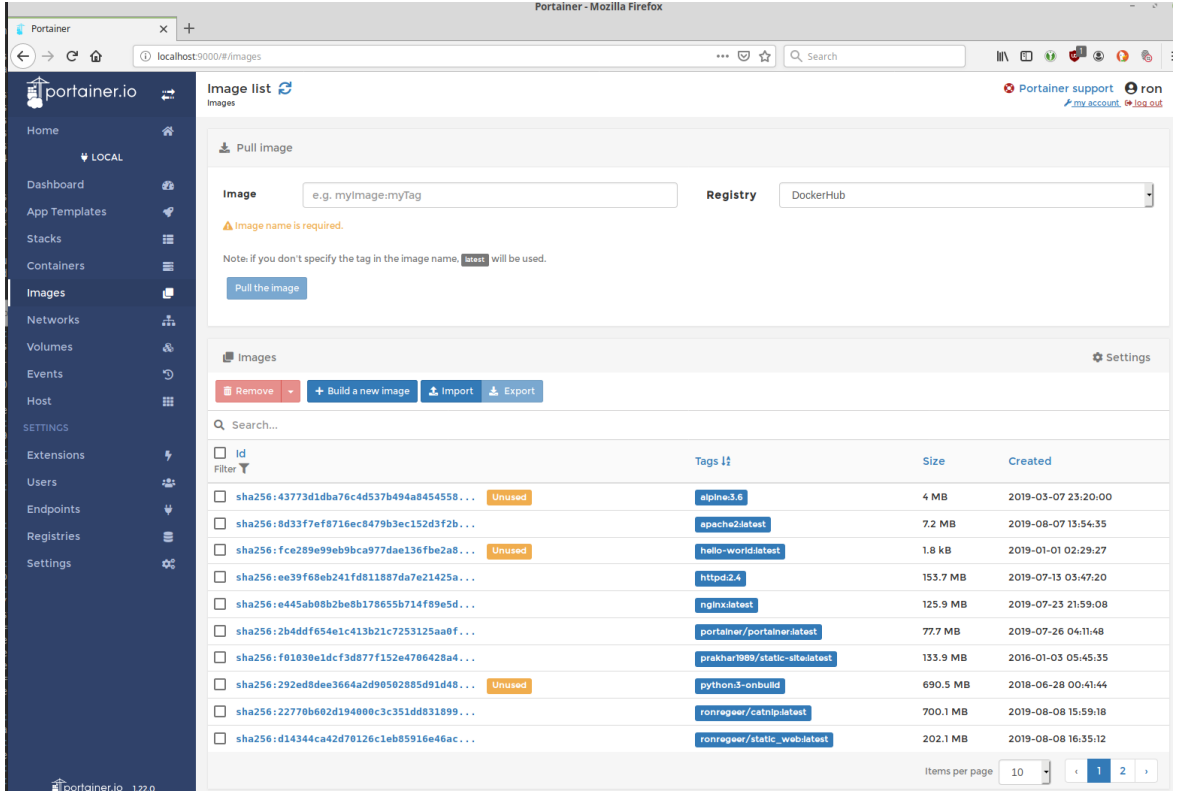
Docker beheren met Portainer

Het dashboard van Portainer



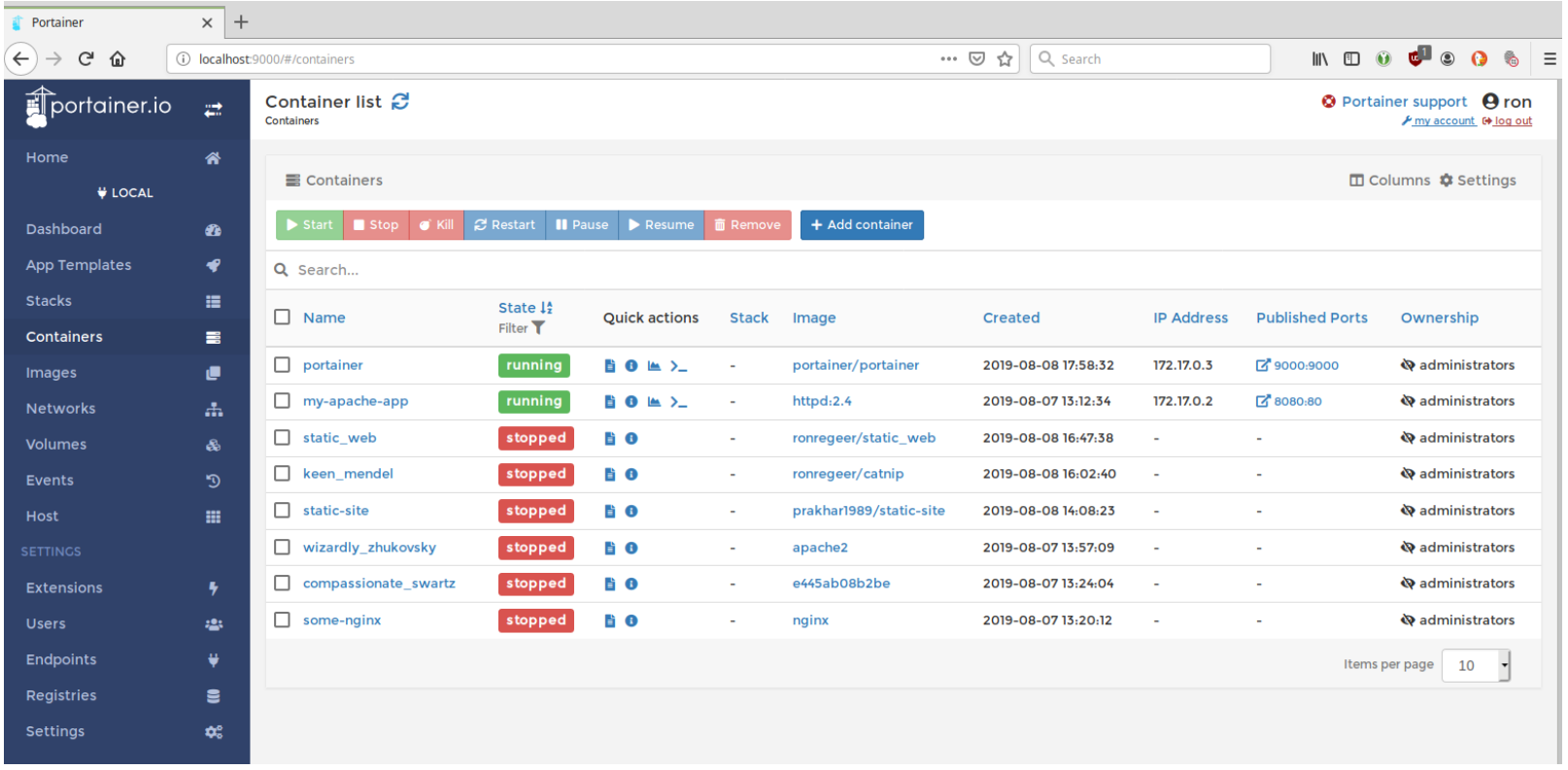
Docker beheren met Portainer

De image list



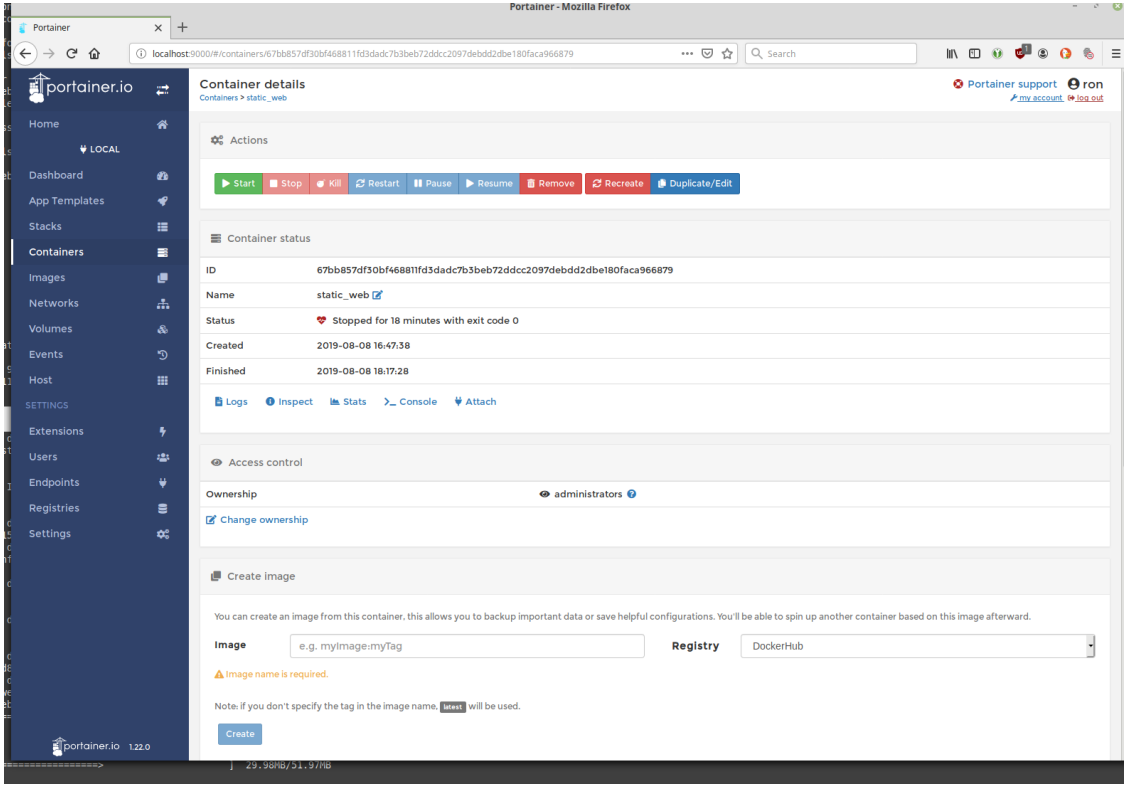
Docker beheren met Portainer

De container list



Docker beheren met Portainer

Details van een container



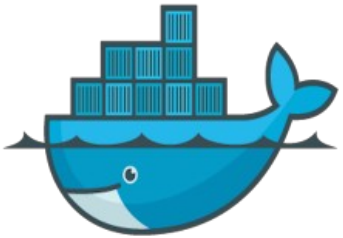
Dit was het dan...

Dit is het einde van mijn presentatie over Docker. Ik ben me bewust van het feit dat er nog heel veel meer te vertellen valt. Mijn doel van deze presentatie was om een tipje van de sluier op te lichten van de mogelijkheden van containerisatie en wat het ongeveer inhoudt. Verder een duwtje in de rug om er zelf mee te experimenteren.

Het is een veelbelovende techniek die naar mijn idee steeds vaker zal worden toegepast. Dan is het als computeraar best leuk om daar meer vanaf te weten. Ook leuk om het toe te passen op je eigen server(s).

Dank voor jullie aandacht.

Ron Regeer.



Presentatie staat op mijn site https://microtux.nl/linux-ig-presentaties/
of kijk op de site van HCC Apeldoorn waar hij binnenkort verschijnt.