

Systems Advanced Docker Containers

Processen uitvoeren in
running containers



Elfde-Liniestraat 24, 3500 Hasselt, www.pxl.be



Processen uitvoeren in running containers

`docker exec [OPTIONS] CONTAINER COMMAND [ARG...]`

Het commando `docker exec` voert een nieuw commando uit in een running container.

Het commando dat met `docker exec` wordt gestart, draait alleen terwijl het primaire proces van de container (PID 1) runt, en het wordt niet opnieuw gestart als de container opnieuw wordt gestart.

COMMAND runt in de standaard directory van de container. Als de onderliggende image een aangepaste directory heeft die is opgegeven met de `WORKDIR` instructie in het Dockerfile, dan wordt die gebruikt.

COMMAND moet een executable zijn, een chained of een quoted commando werkt niet.

Voorbeeld:

- `docker exec -it my_container "echo a && echo b"` zal niet werken,
- maar `docker exec -it my_container sh -c "echo a && echo b"` wel.

Processen uitvoeren in running containers

Begin eerst met een container.

- `docker run --name alpine_sh --rm -it alpine sh`

Dit zal een container met de naam `alpine_sh` aanmaken en een shell sessie starten. Laat de container open staan, start een nieuwe terminal en voer daar een commando uit op de running container.

- `docker exec alpine_sh touch /tmp/testje`

Check in de originele shell sessie of de file er is. Doe dat nog eens met `testje2`.

Voer vervolgens, in de 2de terminal, een 2de interactieve bash-shell uit op de running container.

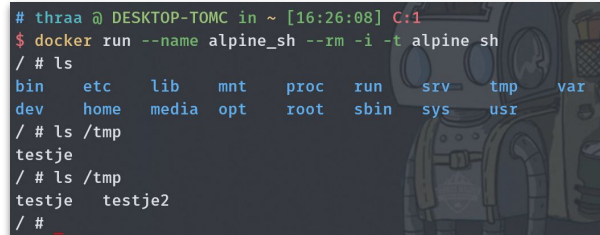
- `docker exec -it alpine_sh sh`
- `exit`

Stel vervolgens in de shell sessie een omgevingsvariabele in, die enkel actief is in die shell sessie

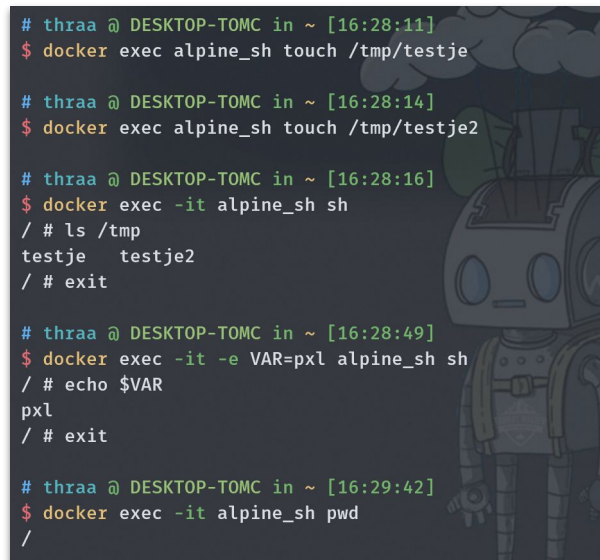
- `docker exec -it -e VAR=pxl alpine_sh sh`
- `exit`

Toon de environment variabelen van de running container

- `docker exec alpine_sh env`



```
# thraa @ DESKTOP-TOMC in ~ [16:26:08] C:1
$ docker run --name alpine_sh --rm -i -t alpine sh
/ # ls
bin    etc    lib    mnt    proc   run    srv    tmp    var
dev    home   media  opt    root   sbin   sys    usr
/ # ls /tmp
testje
/ # ls /tmp
testje  testje2
/ #
```



```
# thraa @ DESKTOP-TOMC in ~ [16:28:11]
$ docker exec alpine_sh touch /tmp/testje2

# thraa @ DESKTOP-TOMC in ~ [16:28:14]
$ docker exec alpine_sh touch /tmp/testje2

# thraa @ DESKTOP-TOMC in ~ [16:28:16]
$ docker exec -it alpine_sh sh
/ # ls /tmp
testje  testje2
/ # exit

# thraa @ DESKTOP-TOMC in ~ [16:28:49]
$ docker exec -it -e VAR=pxl alpine_sh sh
/ # echo $VAR
pxl
/ # exit

# thraa @ DESKTOP-TOMC in ~ [16:29:42]
$ docker exec -it alpine_sh pwd
/
```

Een shell verkrijgen in een container

We proberen een shell te verkrijgen in een container die als PID1 een app draait.

De meeste containers zullen geen **bash** of **ssh** draaien als PID1, maar wel een app waarvoor de container is opgestart.

- We starten eerst een container die 300 keer pingt naar de DNS van Google
 - **docker run -d alpine ping -c 300 8.8.8.8**

Proces sh uitvoeren in een container:

```
docker exec -it <CONTAINER> sh
```

- verlaten met **exit**
 - dit sluit de container niet af!

Ping live bekijken:

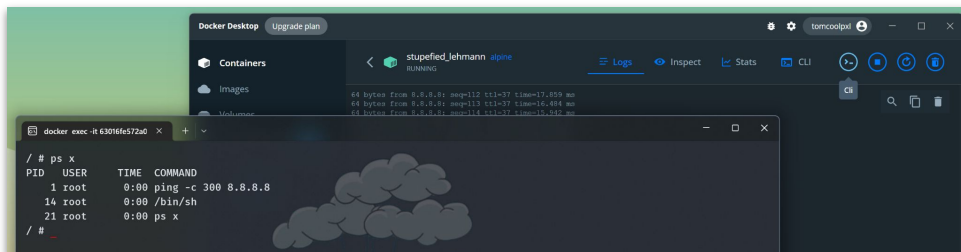
```
docker logs --follow <CONTAINER>
```

```
# thraa @ DESKTOP-TOMC in ~ [23:25:01]
$ docker run -d alpine ping -c 300 8.8.8.8
63016fe572a04d1edde49388590167793b01f0b68d7e0e46c31238a87e2f4bd1

# thraa @ DESKTOP-TOMC in ~ [23:25:36]
$ docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS   NAMES
63016fe572a0   alpine    "ping -c 300 8.8.8.8"    8 seconds ago Up 6 seconds             stupefied_lehmann

# thraa @ DESKTOP-TOMC in ~ [23:25:43]
$ docker exec -it stupefied_lehmann sh
/ # ps x
PID  USER      TIME  COMMAND
1   root      0:00  ping -c 300 8.8.8.8
7   root      0:00  sh
13  root      0:00  ps x
/ # exit

# thraa @ DESKTOP-TOMC in ~ [23:26:29]
$ _
```



Oefeningen

1. Start een nginx container op de achtergrond
 - dan krijg je de cursor ook terug
 - Probeer een shell te verkrijgen in de container met
 - **docker exec**
 - installeer lynx
 - surf naar de website met **lynx localhost**

2. Start een Alpine container, maar zet de prompt als volgt:

`<username>@demo:[<workingdir>] #`

Hints: je kan de environment variable PS1 hiervoor meegeven

Zoek in de manpage van bash naar een regel die begint met PROMPTING

Hieronder kan je de karakters vinden voor de `<username>` en `<working dir>`

Oefeningen

3. Start een ubuntu-image met
 - `docker run -d -it --name <naam> ubuntu`
 - voer het `tree` commando uit
 - `docker exec -it <naam> tree`
 - OCI runtime exec failed: exec failed: container_linux.go:346: starting container process caused "exec: tree": executable file not found in \$PATH": unknown
Er is duidelijk iets kapot. Misschien ontbreekt tree?
 - Ga in de container en fix het probleem.
 - Voer nu het `tree` commando opnieuw uit vanaf de commandoregel.

4. Nu we opgewarmd zijn is het tijd om in een echte container te stappen.
 - De image `pxlsystemsadvanced/simplewebservice:ubuntu` start een container die logs uitvoert naar een bestand. Ga in de container en gebruik `tail -f ./text.log` om de logs te volgen. Elke 10 seconden zal de klok je een "geheime boodschap" sturen.
 - Vind de geheime boodschap.

