Lektionstillfälle 3

”Dockerfile”

Utbildare: Robert Westin

NACKADEMIN

Kort summering av föregående lektion/ev. lektioner

Föregående lektion:

- docker
  - docker container run
    - -d -detach "Run container in background ..."
    - -i "Keep STDIN open even if not attached"
    - -t "--tty Allocate a pseudo-TTY"
  - docker container ls
  - docker container stop
  - docker container rm (--force)
  - docker exec vs docker attach

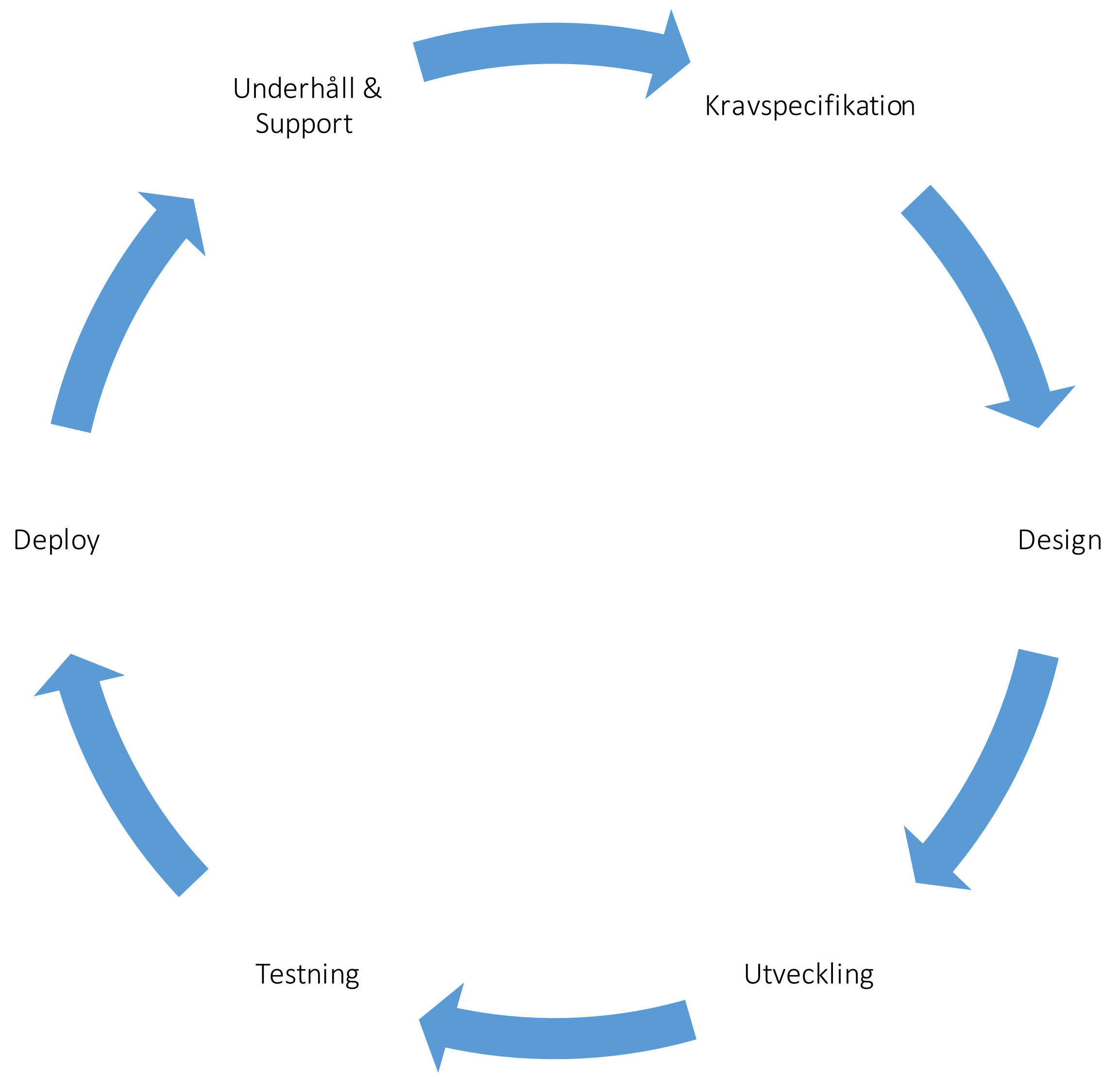
NACKADEMIN

Repetition av veckan

Föregående lektion 1:

- Continuous Integration
- Continuous Delivery & Continuous Deployment
- Python Flask
- GitHub Organization
- GitHub Repository
- GitHub Collaborator
- GitHub Teams
- GitHub Actions
- GitHub Classroom

En Applikations livscykel



CI - Continuous Integration

- Versionshantering av kod
- Automatisera "bygget"
  - Vad är ett kodbygge?
  - Vad är resultatet av ett bygge?
- Bygg varje ny kod ändring
- Automatisera tester
- Automatisera gemensamma regler & statisk kodanalys
  - Kodstil, komplexitet?
- Gick bygget bra eller dåligt? Resultat synligt!

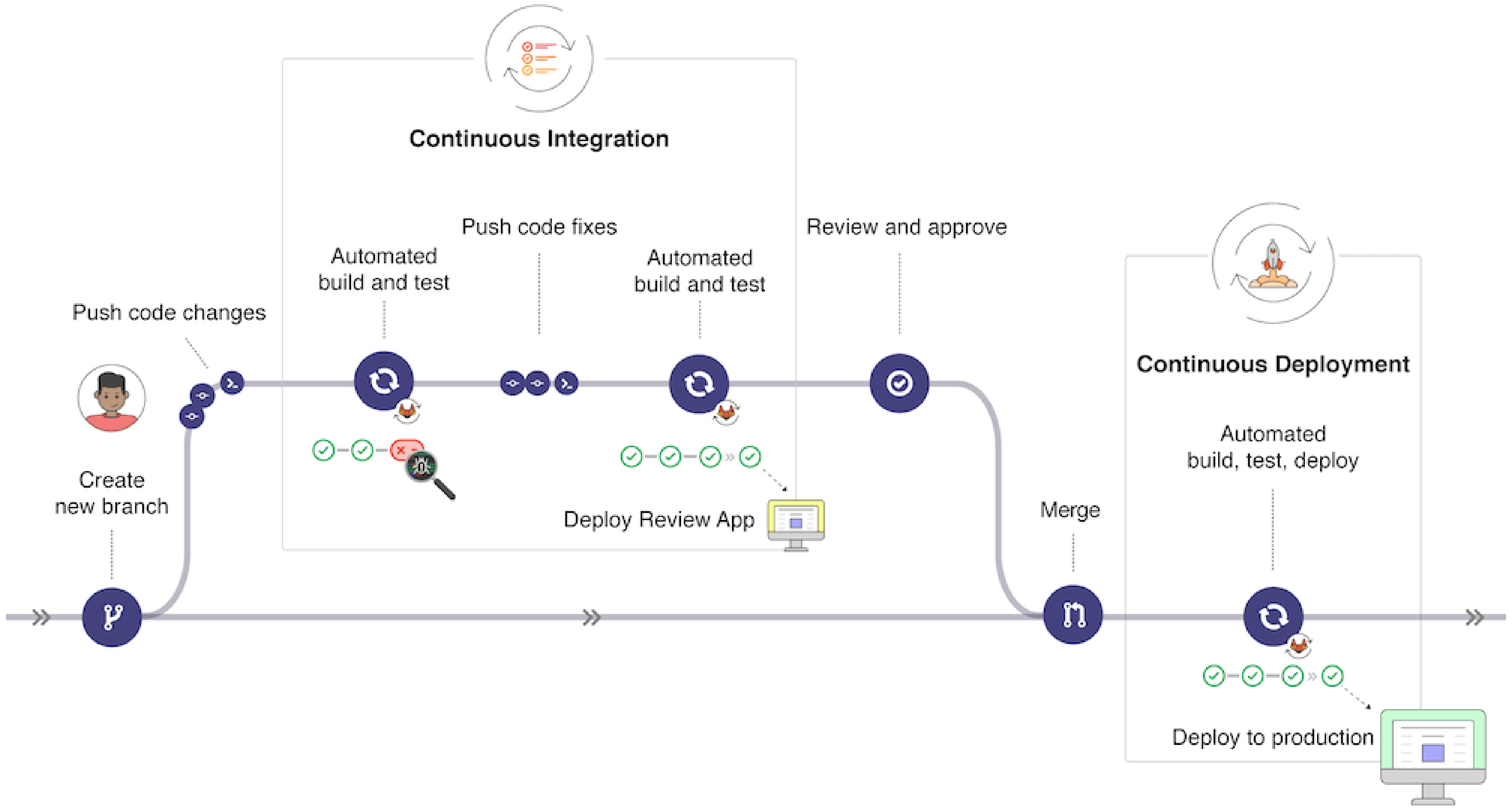
NACKADEMIN

CD - Continuous Delivery varför?

- Fokus på att alltid vara klar för release
- Nästan alla steg är automatiserade
- Maximera utvecklarnas tid för utveckling istället för administration
- Automatisera bort repetitiva uppgifter i organisationen
- Ta bort den mänskliga faktorn
- Identifiera och ta bort saker som senarelägger "Time to market"
- Tillförlitliga leveranser med jämn kvalitet
- Undvik personberoenden, release Kim är sjuk idag?

NACKADEMIN

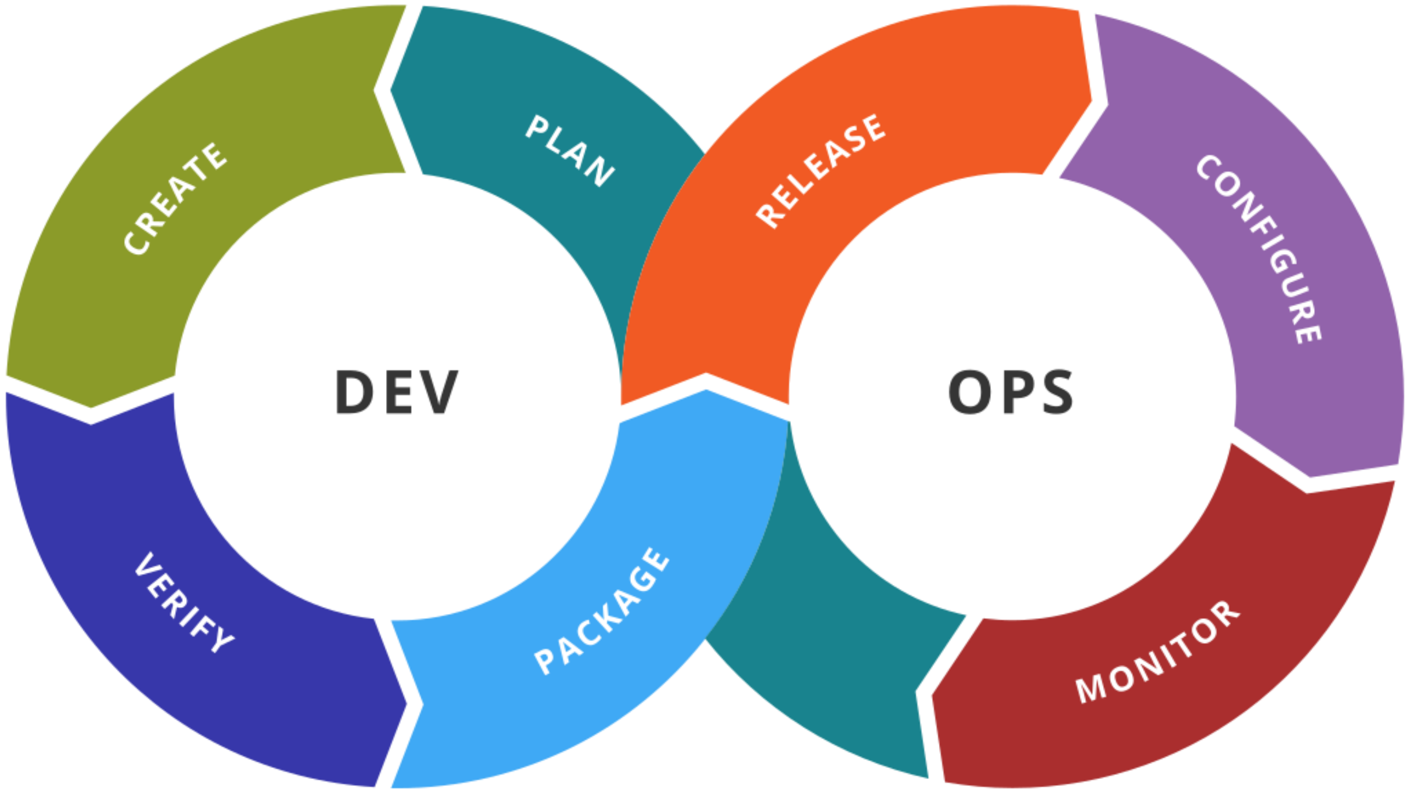
Continuous Deployment



NACKADEMIN

Källa: https://docs.gitlab.com/ee/ci/introduction/

DevOps – Vart är CI? CD?



Källa: Wikimedia - Kharnagy

Python Flask

https://github.com/microsoft/python-sample-vscode-flask-tutorial

NACKADEMIN

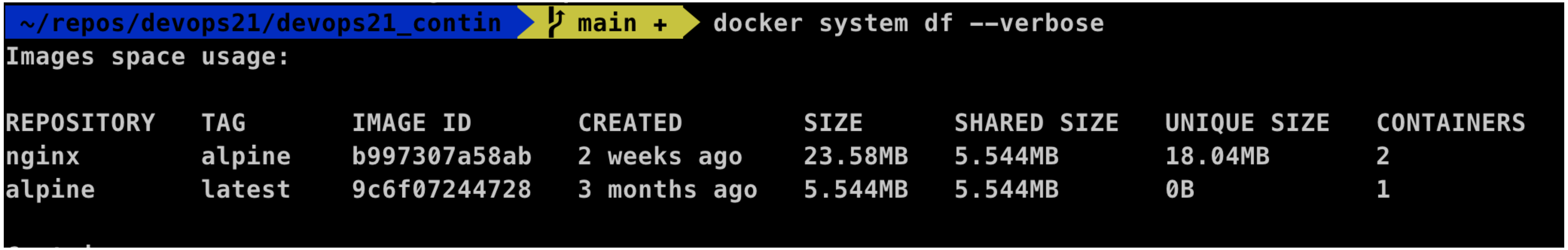
Repetition av veckan

Föregående lektion 2:

- Docker
- Image
- Container
- Lagring icke-persistent vs persistent
- Nätverk i Docker

NACKADEMIN

Docker – Shared vs Unique size



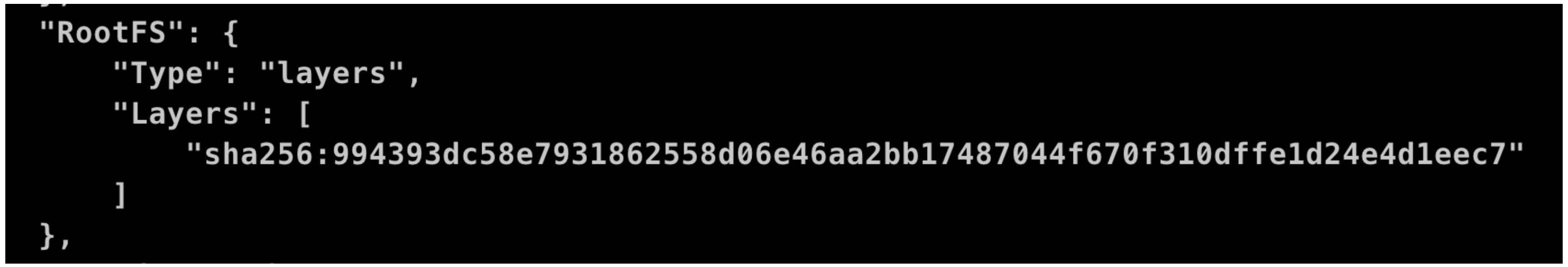
```
~/repos/devops21/devops21_contin  main +  docker system df --verbose
Images space usage:

REPOSITORY    TAG         IMAGE ID        CREATED          SIZE        SHARED SIZE    UNIQUE SIZE    CONTAINERS
nginx         alpine      b997307a58ab    2 weeks ago      23.58MB     5.544MB        18.04MB        2
alpine        latest      9c6f07244728    3 months ago     5.544MB     5.544MB        0B             1
```

- SHARED SIZE is the amount of space that an image shares with another one (i.e. their common data)
- UNIQUE SIZE is the amount of space that is only used by a given image
- SIZE is the virtual size of the image, it is the sum of SHARED SIZE and UNIQUE SIZE

NACKADEMIN

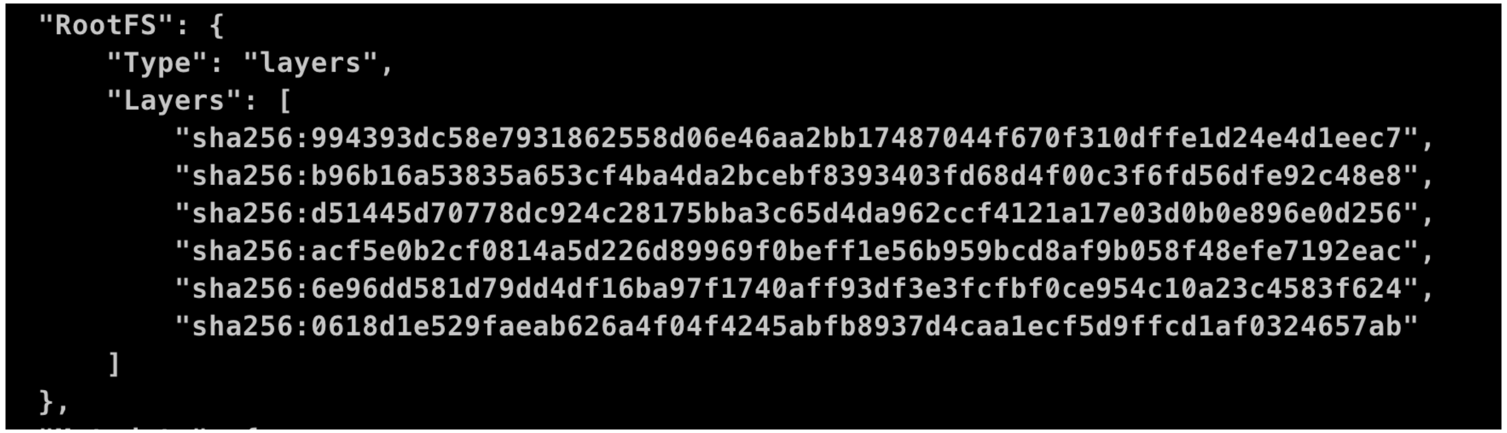
Docker – image inspect

- docker image inspect alpine



```
"RootFS": {
    "Type": "layers",
    "Layers": [
        "sha256:994393dc58e7931862558d06e46aa2bb17487044f670f310dffe1d24e4d1eec7"
    ]
},
```

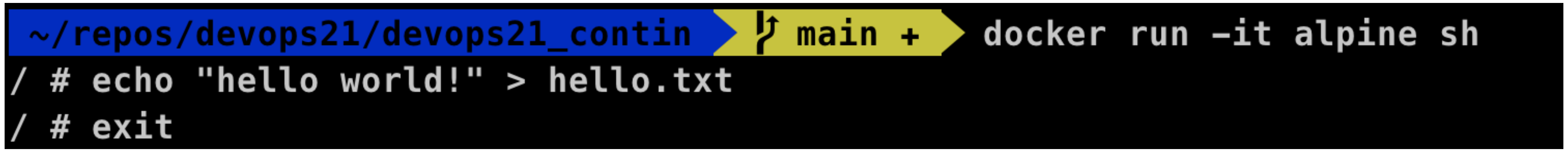
- docker image inspect nginx:alpine



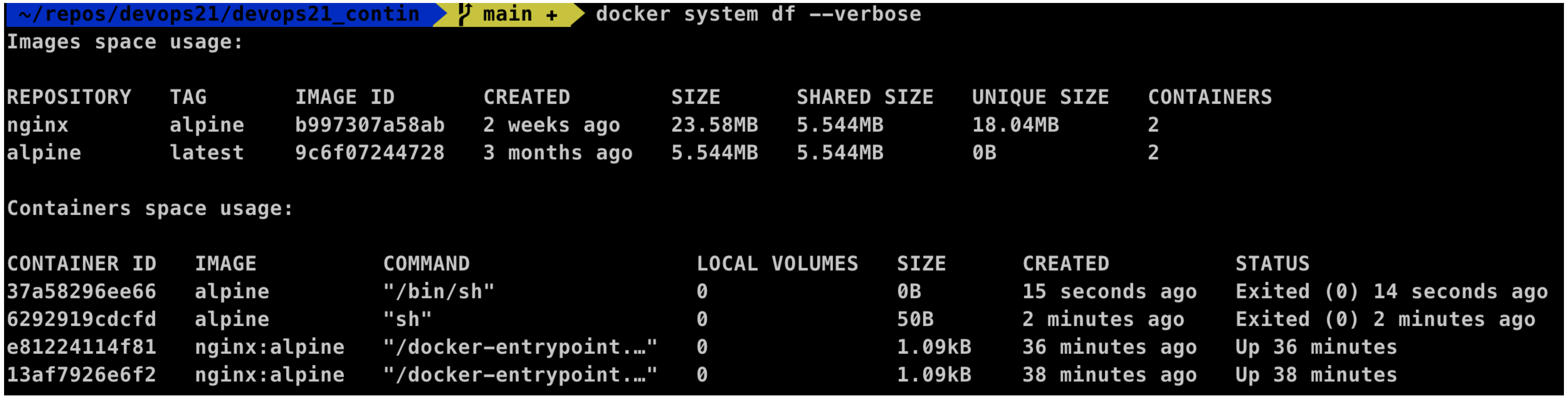
```
"RootFS": {
    "Type": "layers",
    "Layers": [
        "sha256:994393dc58e7931862558d06e46aa2bb17487044f670f310dffe1d24e4d1eec7",
        "sha256:b96b16a53835a653cf4ba4da2bcebf8393403fd68d4f00c3f6fd56dfe92c48e8",
        "sha256:d51445d70778dc924c28175bba3c65d4da962ccf4121a17e03d0b0e896e0d256",
        "sha256:acf5e0b2cf0814a5d226d89969f0beff1e56b959bcd8af9b058f48efe7192eac",
        "sha256:6e96dd581d79dd4df16ba97f1740aff93df3e3fcfbf0ce954c10a23c4583f624",
        "sha256:0618d1e529faeab626a4f04f4245abfb8937d4caa1ecf5d9ffcd1af0324657ab"
    ]
},
```

NACKADEMIN

Docker – Container



```
~/repos/devops21/devops21_contin  main +  docker run -it alpine sh
/ # echo "hello world!" > hello.txt
/ # exit
```



```
~/repos/devops21/devops21_contin  main +  docker system df --verbose
Images space usage:

REPOSITORY   TAG       IMAGE ID       CREATED        SIZE      SHARED SIZE   UNIQUE SIZE   CONTAINERS
nginx        alpine    b997307a58ab   2 weeks ago    23.58MB   5.544MB       18.04MB       2
alpine       latest    9c6f07244728   3 months ago   5.544MB   5.544MB       0B            2

Containers space usage:

CONTAINER ID   IMAGE          COMMAND                  LOCAL VOLUMES   SIZE      CREATED          STATUS
37a58296ee66   alpine         "/bin/sh"                0               0B        15 seconds ago   Exited (0) 14 seconds ago
6292919cdcfd   alpine         "sh"                     0               50B       2 minutes ago    Exited (0) 2 minutes ago
e81224114f81   nginx:alpine   "/docker-entrypoint.…"   0               1.09kB    36 minutes ago   Up 36 minutes
13af7926e6f2   nginx:alpine   "/docker-entrypoint.…"   0               1.09kB    38 minutes ago   Up 38 minutes
```

NACKADEMIN

Storage Drivers vs Docker volumes

Docker storage drivers -

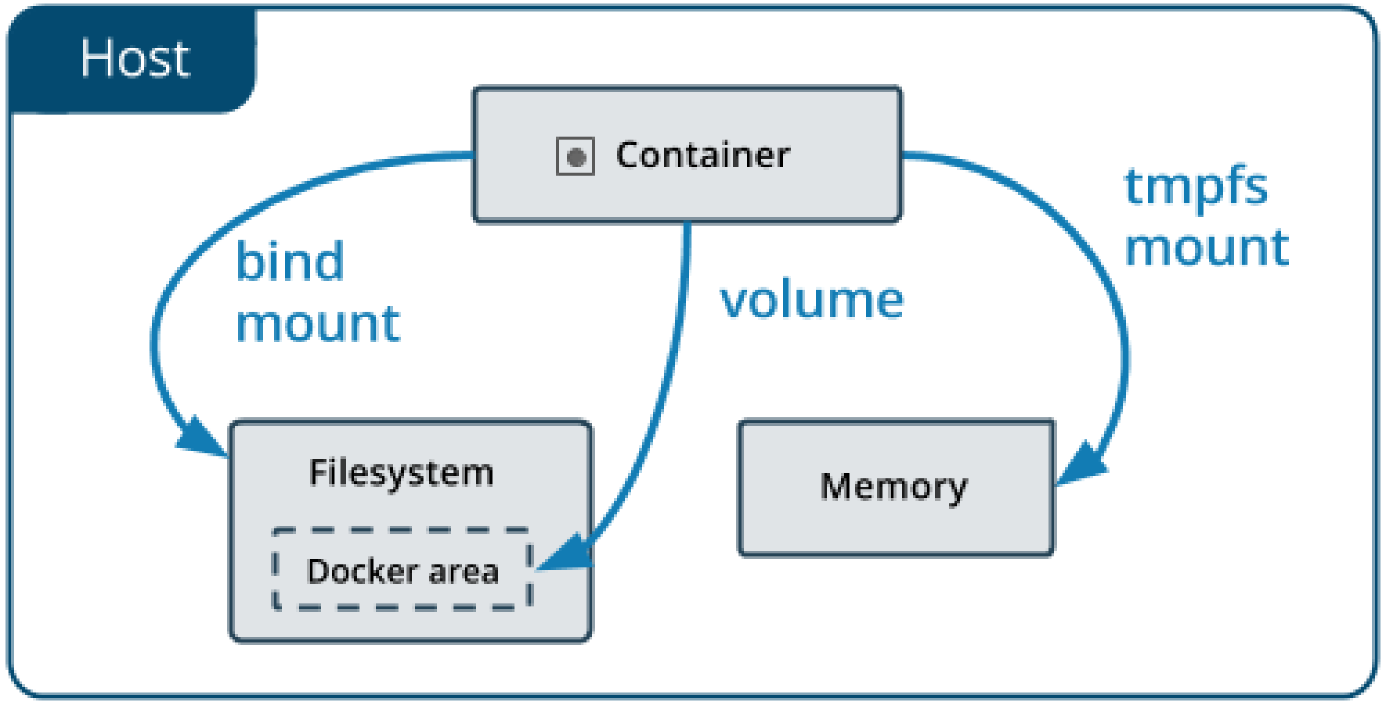
- Image lager
- Skriv lager för container (inte persistent utan container)
- Runtime data
- Optimerat för att använda lagringsutrymmet effektivt
- Sämre skrivprestanda (COW)

Docker volumes

- Intensivt skrivande av data
- Persistent utanför container
- Delat mellan containers

Källa: https://docs.docker.com/storage/storagedriver/

Docker välj rätt mount



Källa: https://docs.docker.com/storage/

NACKADEMIN

Docker välj rätt mount

- **Volumes**
  - Sparas i värdens filsystem
  - Managerat av docker *Docker* (/var/lib/docker/volumes/ på Linux).
  - Låt inte andra än docker modifiera filerna
  - Bästa sättet att persistera data i docker
- **Bind mounts**
  - Filer eller foldrar som sparas varsomhelst på värdens filsystem
  - Både docker och värden kan modifiera filerna
- **tmpfs mounts**
  - Sparas i värdens minne
  - Sparas inte på värdens filsystem
  - Om containern stannar så försvinner filerna

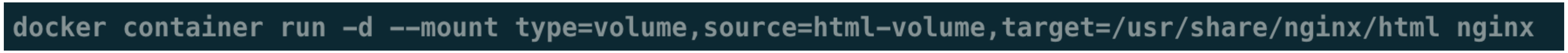
Källa: https://docs.docker.com/storage/

NACKADEMIN

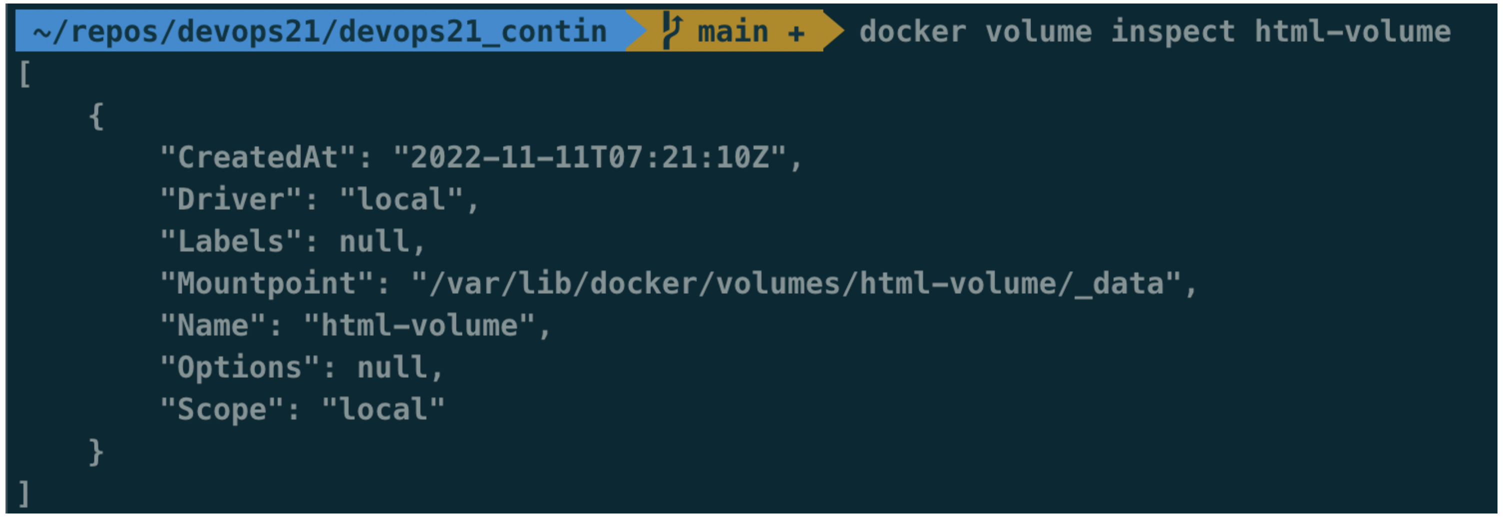
Docker bind volume

Se https://docs.docker.com/storage/volumes/#create-and-manage-volumes för mer exempel

Exempel:



```
docker container run -d --mount type=volume,source=html-volume,target=/usr/share/nginx/html nginx
```



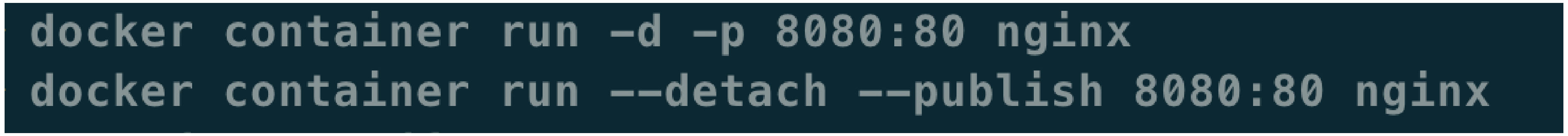
```
~/repos/devops21/devops21_contin  main +  docker volume inspect html-volume
[
    {
        "CreatedAt": "2022-11-11T07:21:10Z",
        "Driver": "local",
        "Labels": null,
        "Mountpoint": "/var/lib/docker/volumes/html-volume/_data",
        "Name": "html-volume",
        "Options": null,
        "Scope": "local"
    }
]
```

NACKADEMIN

Docker container nätverk

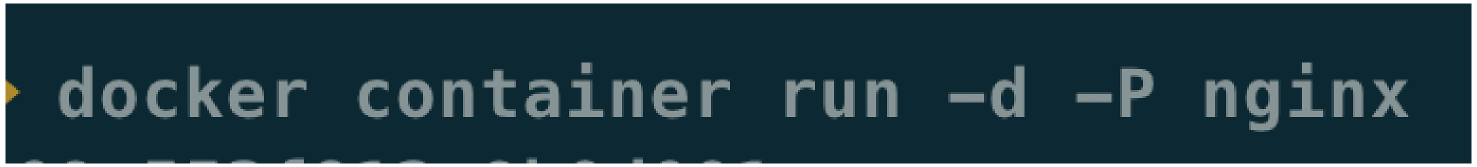
Publish gör så att portar kan mappas mot din host

- I exempel mappas porten 8080 för mot containers port 80
- Publish används för att utveckla och testa dina egna applikationer



```
docker container run -d -p 8080:80 nginx
docker container run --detach --publish 8080:80 nginx
```

-P ger dockervald port



```
docker container run -d -P nginx
```

NACKADEMIN

Docker container nätverk

https://docs.docker.com/network/

Bridge

Host

Overlay

Ipvlan

Macvlan

None

Plugins (https://docs.docker.com/engine/extend/plugins_services/#network-plugins)

Docker bridge nätverk

Bridge

- Standardval driver
- Default nätverks driver
- Bra för ensamma containers

Läs mer: https://docs.docker.com/network/bridge/

Labb: https://docs.docker.com/network/network-tutorial-standalone/

NACKADEMIN

Lektionstillfällets mål och metod

Mål med lektionen:

- Docker fortsättning
- Dockerfile för att bygga images

Lektionens arbetsmetod/er:

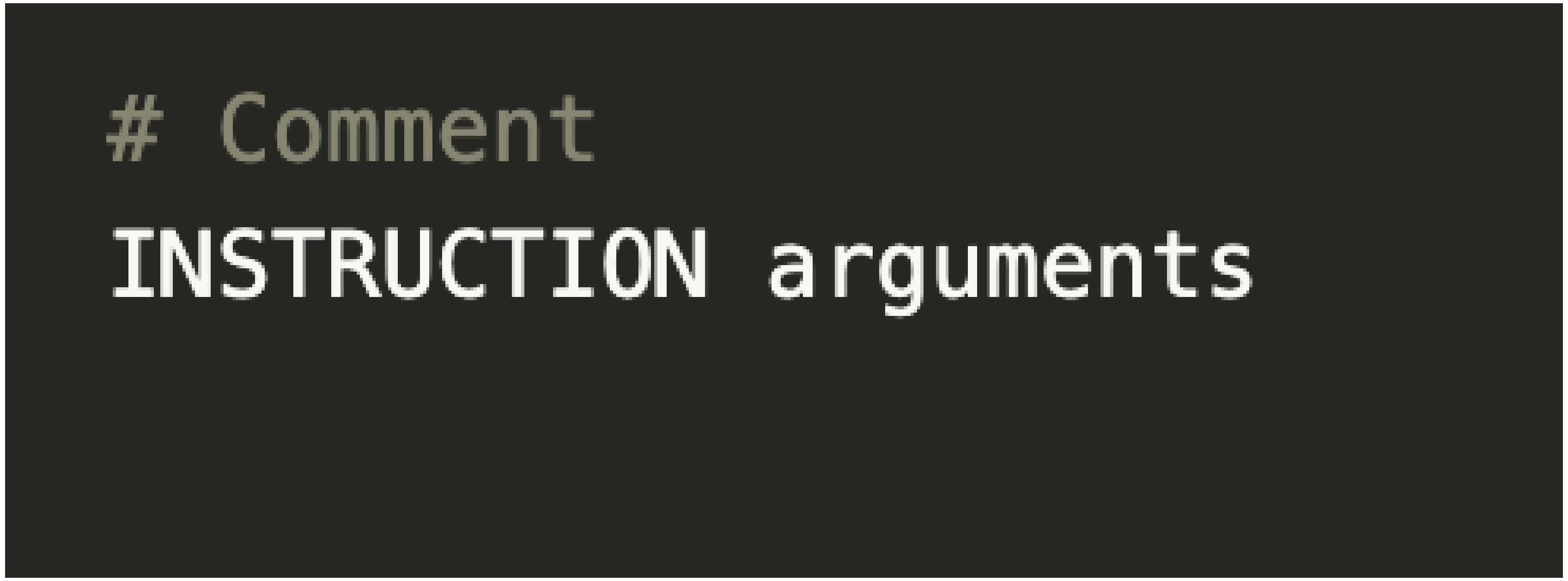
- Föreläsning
- Labb och övning

NACKADEMIN

Begreppsgenomgång

- Dockerfile
  - Instructions
  - Layers
  - Best practice
  - Multi-Stage
- .dockerignore
- Registry (login, build, push)
- nginx
  - .html
  - .conf
- Python

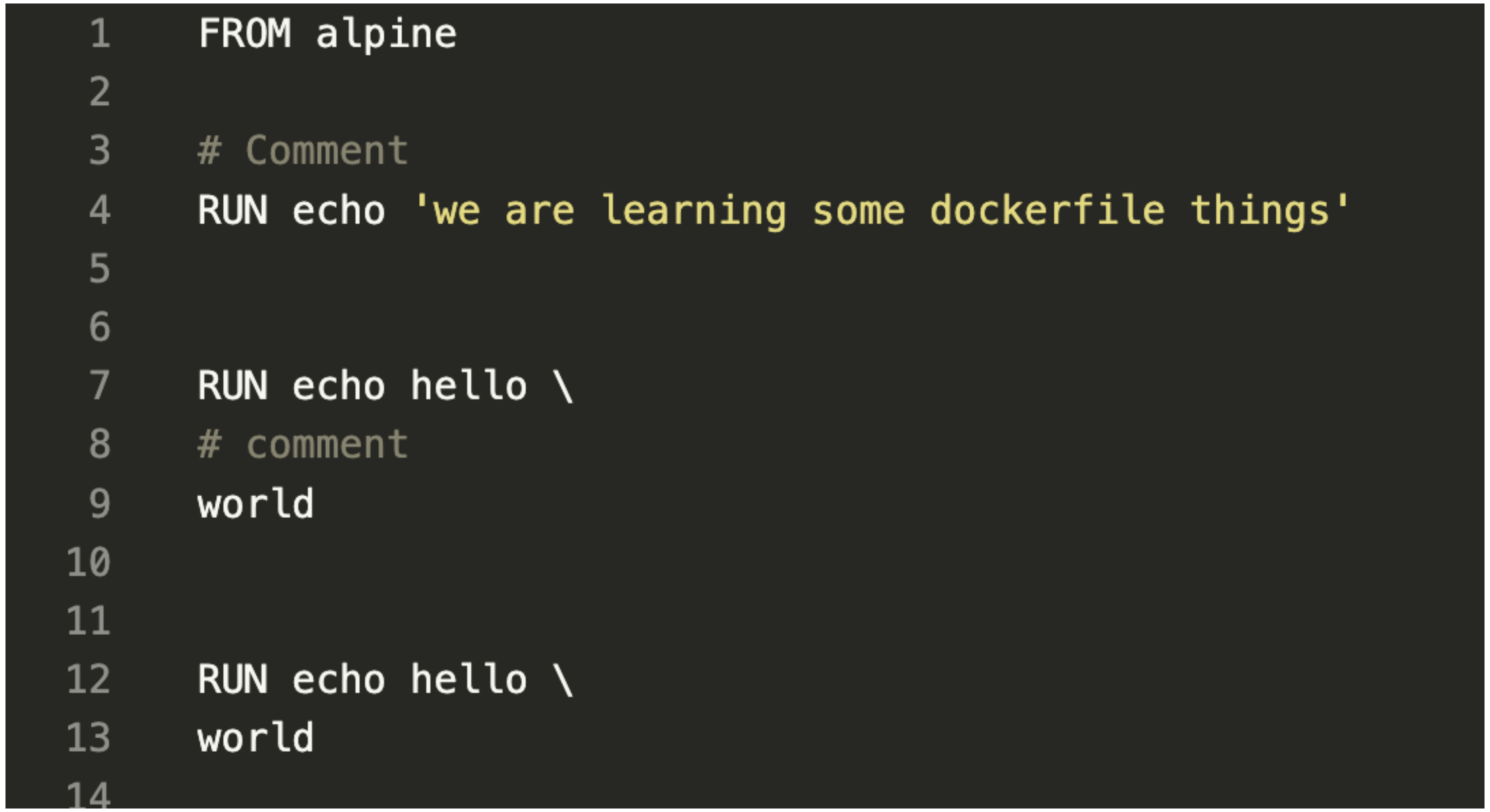
Dockerfile Format



```
# Comment
INSTRUCTION arguments
```

NACKADEMIN

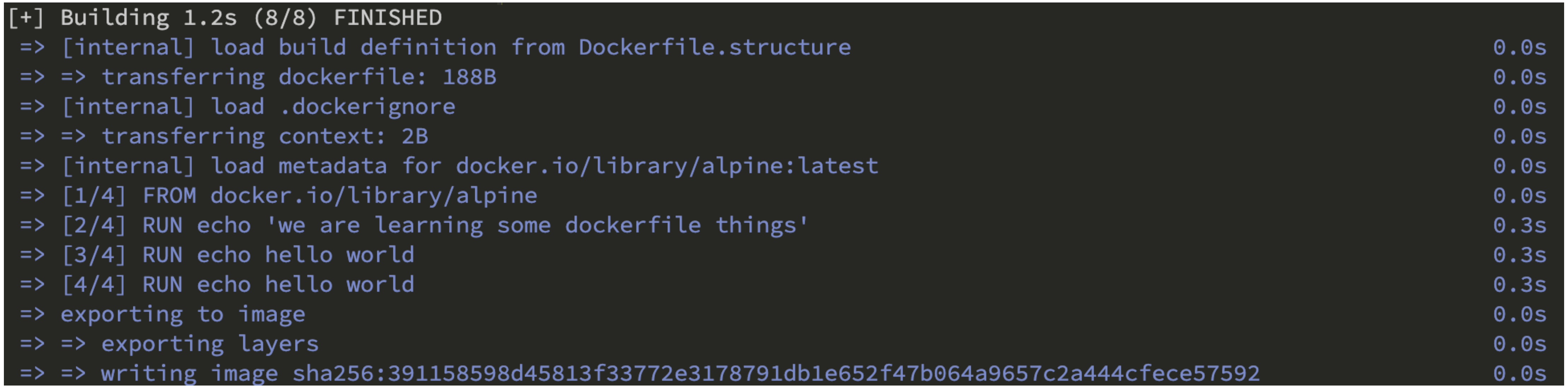
Dockerfile Format



```dockerfile
FROM alpine

# Comment
RUN echo 'we are learning some dockerfile things'


RUN echo hello \
# comment
world


RUN echo hello \
world

```

NACKADEMIN

Dockerfile

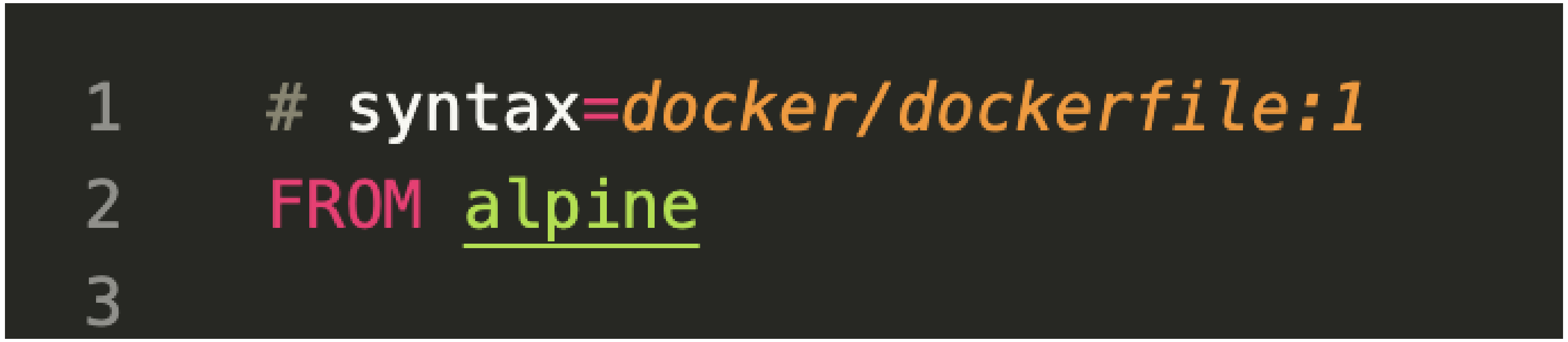


```
[+] Building 1.2s (8/8) FINISHED
 => [internal] load build definition from Dockerfile.structure                                  0.0s
 => => transferring dockerfile: 188B                                                             0.0s
 => [internal] load .dockerignore                                                                0.0s
 => => transferring context: 2B                                                                  0.0s
 => [internal] load metadata for docker.io/library/alpine:latest                                 0.0s
 => [1/4] FROM docker.io/library/alpine                                                          0.0s
 => [2/4] RUN echo 'we are learning some dockerfile things'                                      0.3s
 => [3/4] RUN echo hello world                                                                   0.3s
 => [4/4] RUN echo hello world                                                                   0.3s
 => exporting to image                                                                           0.0s
 => => exporting layers                                                                          0.0s
 => => writing image sha256:391158598d45813f33772e3178791db1e652f47b064a9657c2a444cfece57592     0.0s
```

NACKADEMIN

Dockerfile Parser-directives

- Syntax
  - https://semver.org/
  - https://docs.docker.com/engine/reference/builder/#syntax
  - BuildKit (frontend & backend)
  - Optimerade byggen, egen syntax etc.

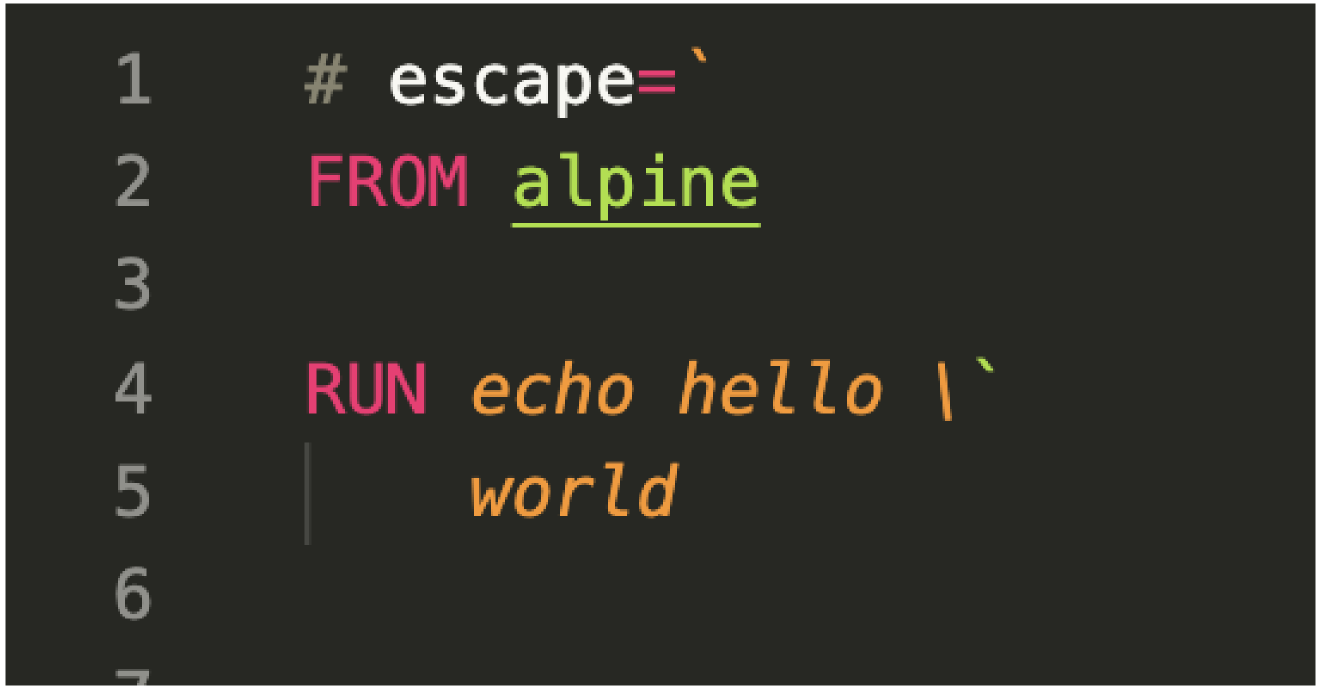


```dockerfile
# syntax=docker/dockerfile:1
FROM alpine

```

NACKADEMIN

Dockerfile Parser-directives

- Escape
  - https://docs.docker.com/engine/reference/builder/#escape
  - Bra för windows användare

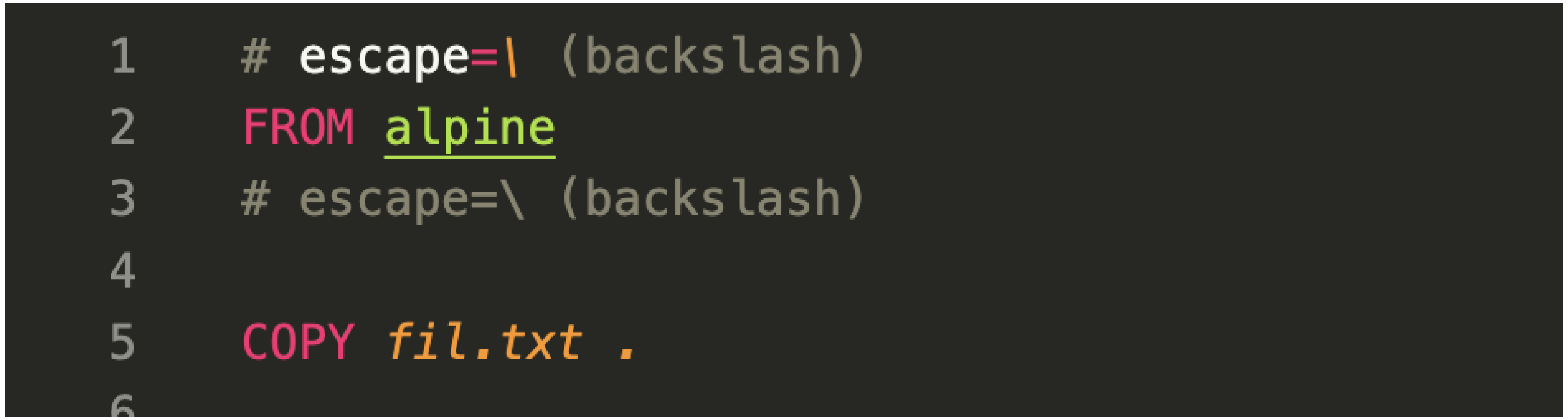


```
# escape=`
FROM alpine

RUN echo hello \`
    world
```

NACKADEMIN

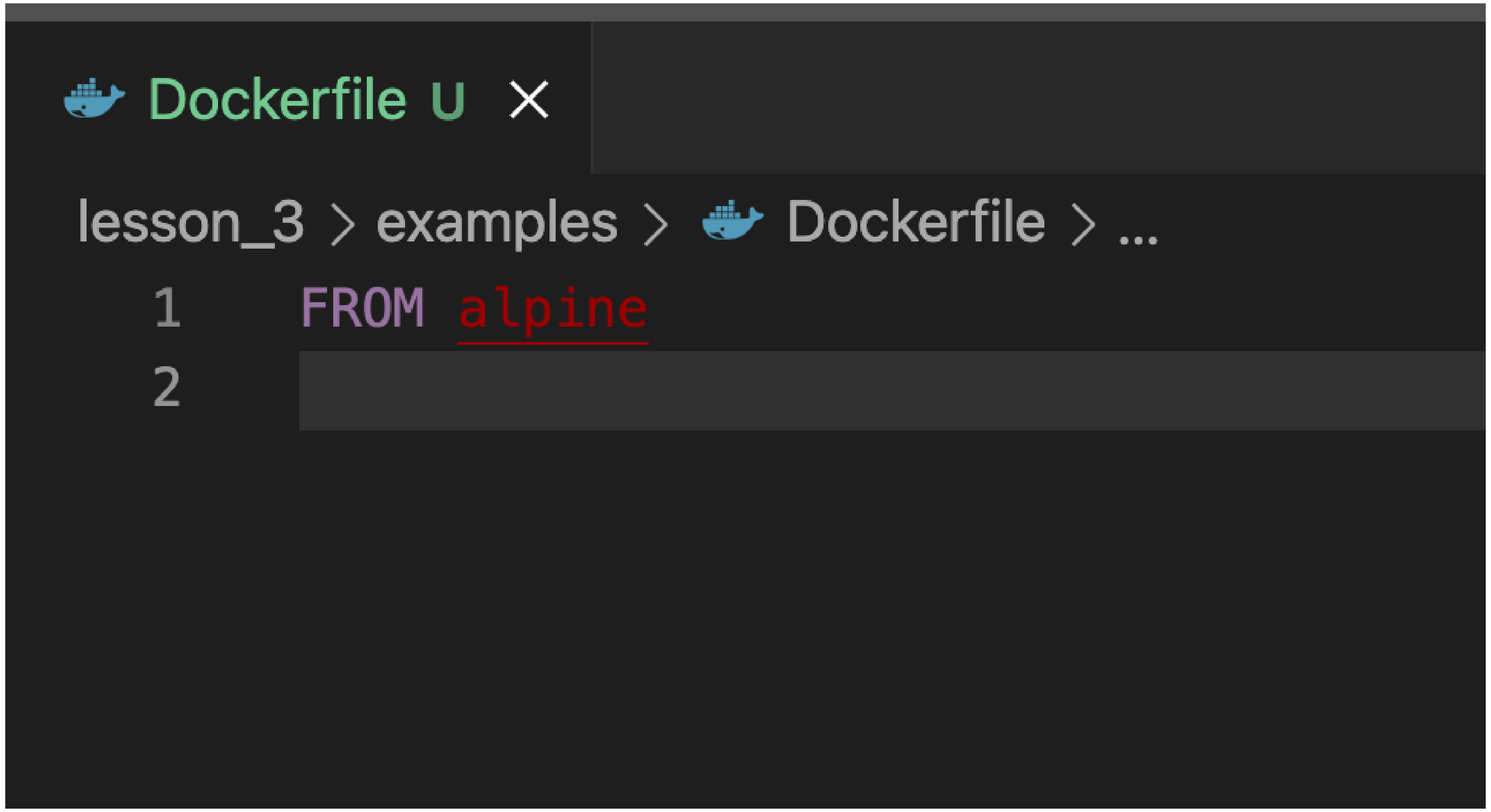
Dockerfile Parser-directives

- Docker läser bara en parser-directive
- Parser-directives ska vara före FROM



```
# escape=\ (backslash)
FROM alpine
# escape=\ (backslash)

COPY fil.txt .
```

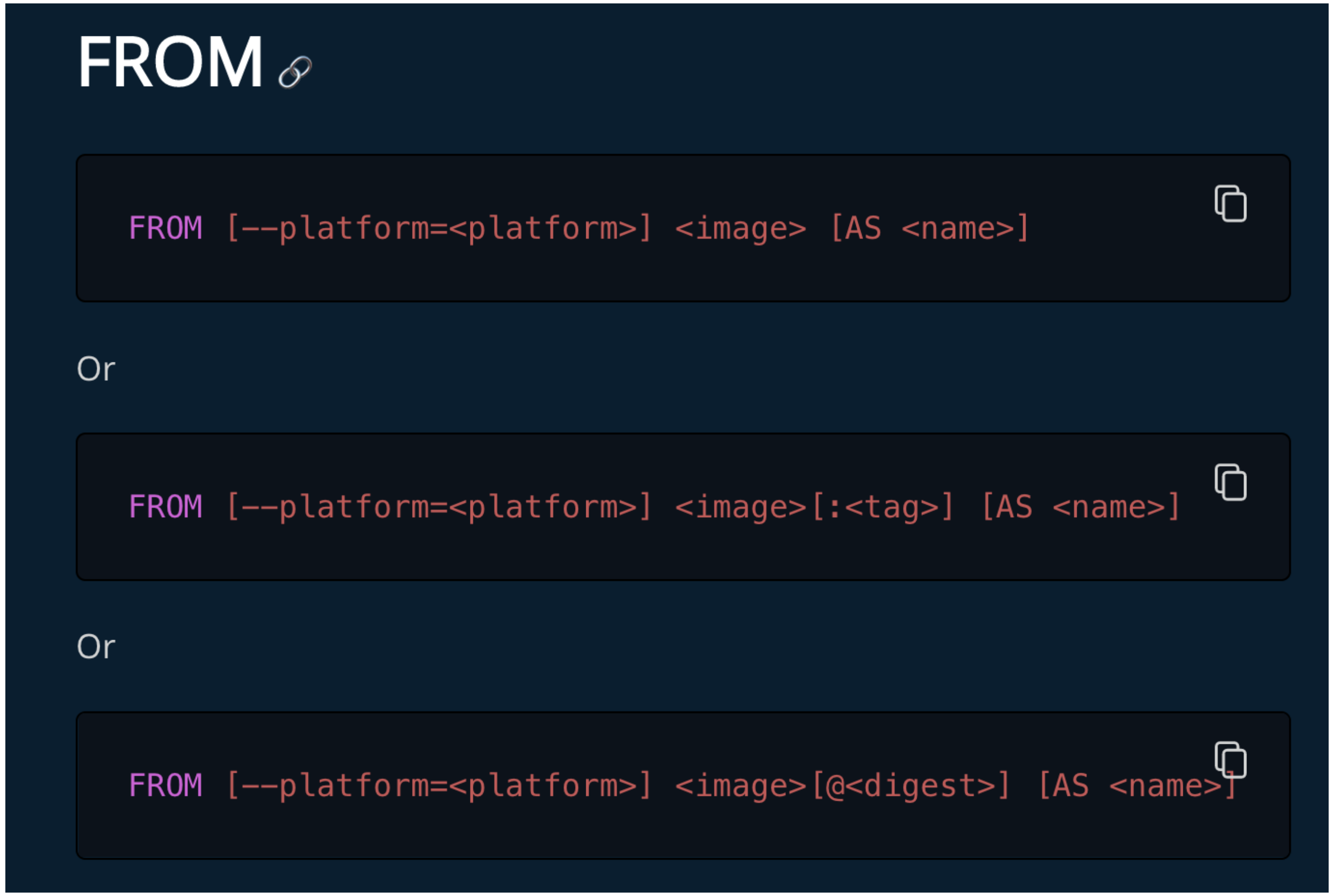
FROM



```
FROM alpine

```

NACKADEMIN

FROM


FROM

```
FROM [--platform=<platform>] <image> [AS <name>]
```

Or

```
FROM [--platform=<platform>] <image>[:<tag>] [AS <name>]
```

Or

```
FROM [--platform=<platform>] <image>[@<digest>] [AS <name>]
```

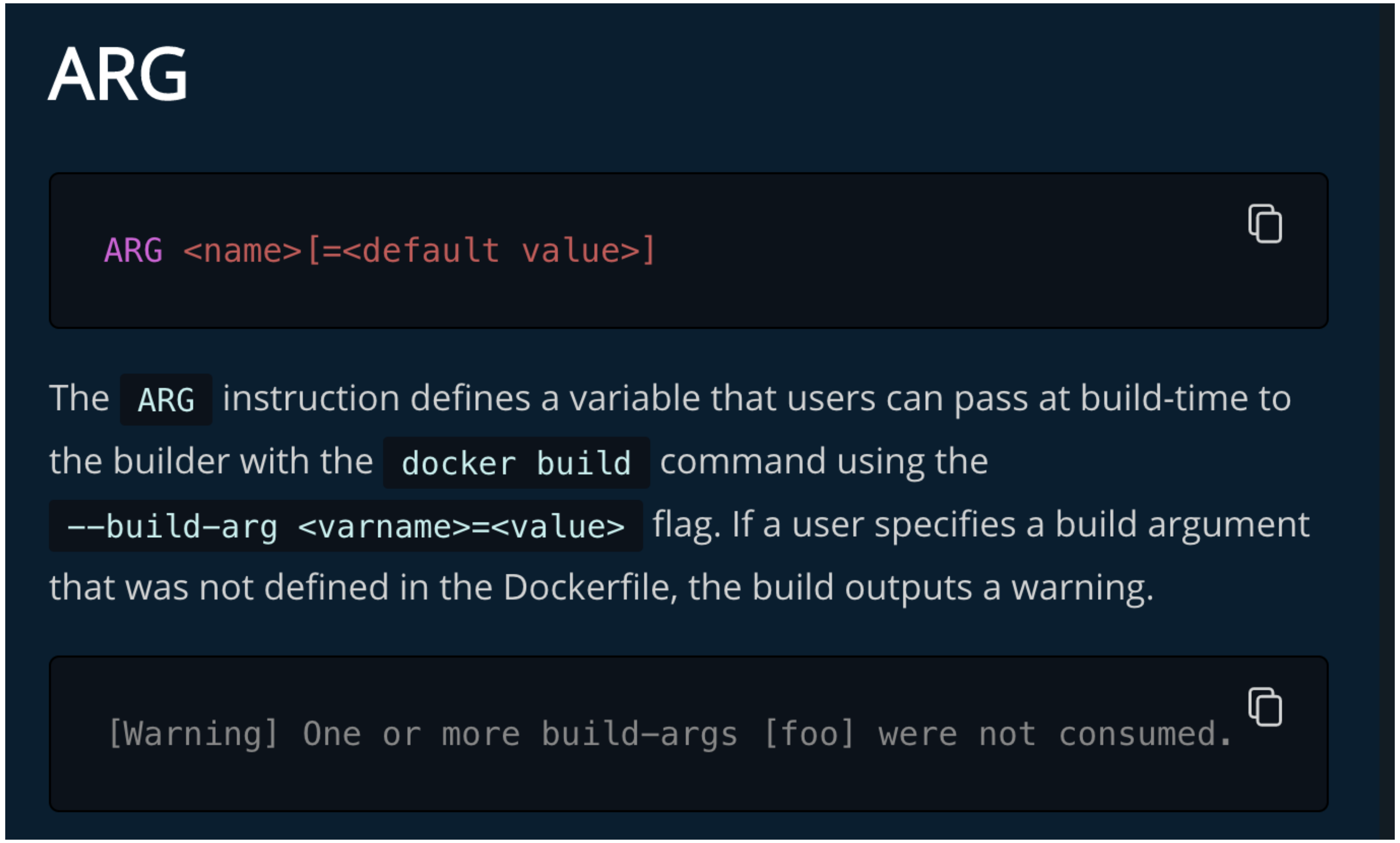
Källa: https://docs.docker.com/engine/reference/builder/#from

NACKADEMIN

FROM

- FROM är ofta först i en fil
  - Om inte parser direktiv finns
  - Endast instruktionen ARG får vara före
- FROM används för att bestämma "base image"
  - Debian
  - Alpine
  - Nginx
- FROM kan användas flera gånger
  - Multi-stage build

ARG



ARG

```
ARG <name>[=<default value>]
```

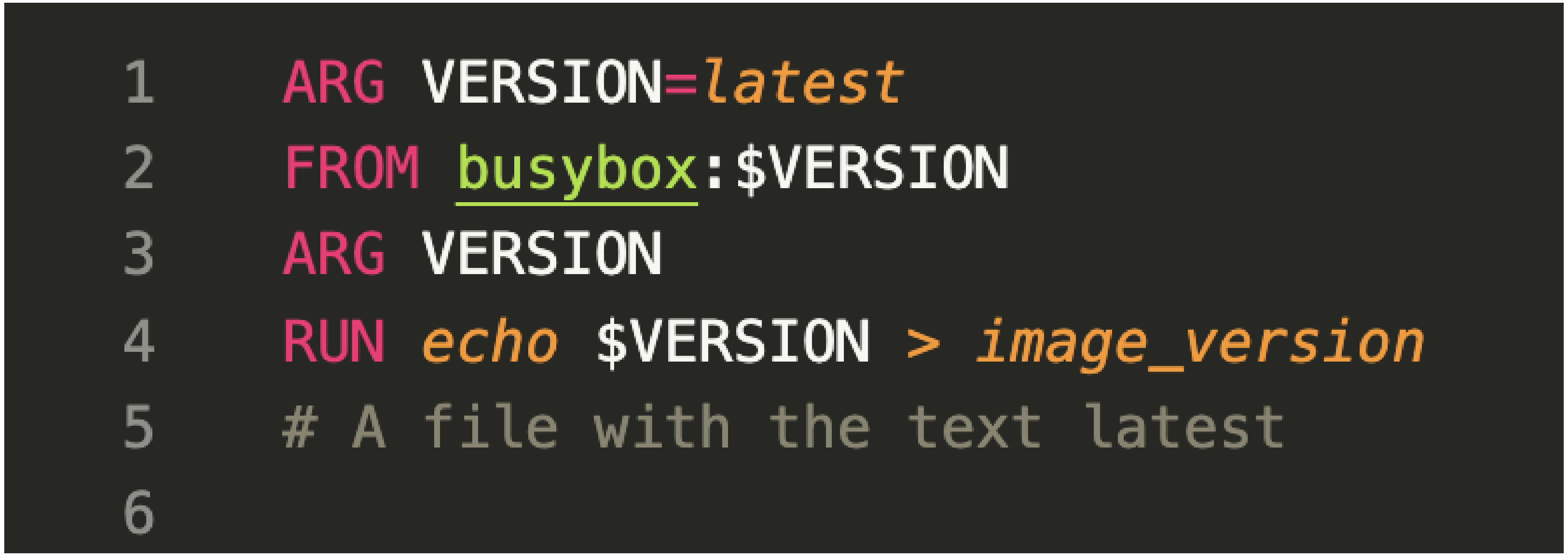
The `ARG` instruction defines a variable that users can pass at build-time to the builder with the `docker build` command using the `--build-arg <varname>=<value>` flag. If a user specifies a build argument that was not defined in the Dockerfile, the build outputs a warning.

```
[Warning] One or more build-args [foo] were not consumed.
```

Källa: https://docs.docker.com/engine/reference/builder/#arg

NACKADEMIN

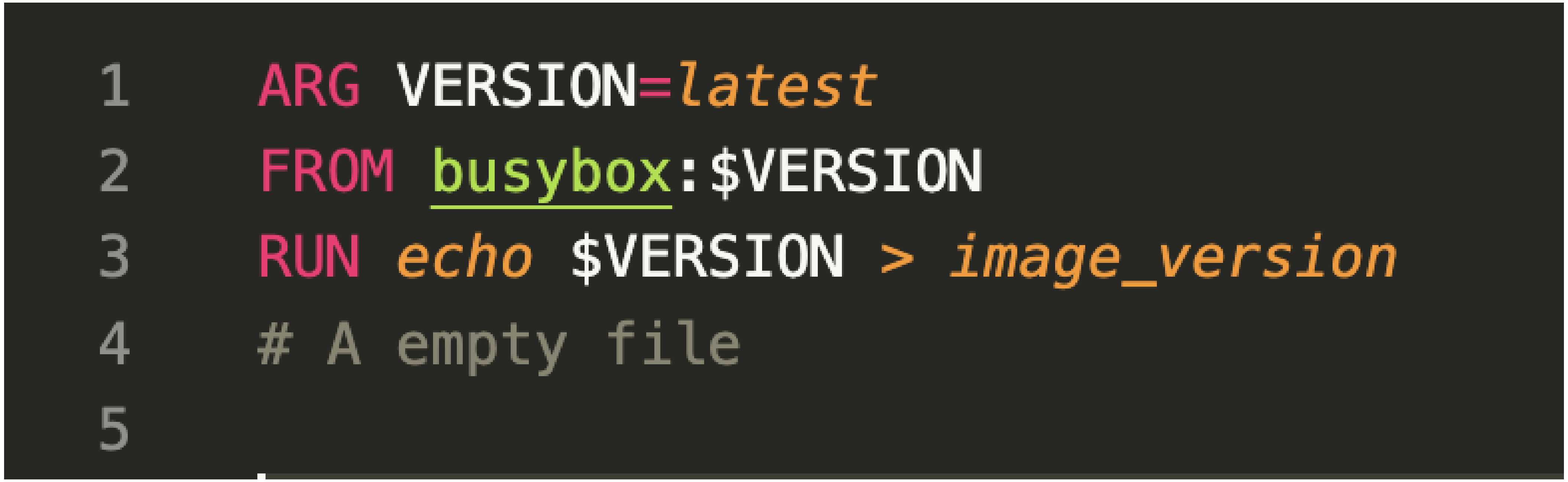
ARG och FROM



```
ARG VERSION=latest
FROM busybox:$VERSION
ARG VERSION
RUN echo $VERSION > image_version
# A file with the text latest

```

NACKADEMIN

ARG och FROM



```dockerfile
ARG VERSION=latest
FROM busybox:$VERSION
RUN echo $VERSION > image_version
# A empty file

```

NACKADEMIN

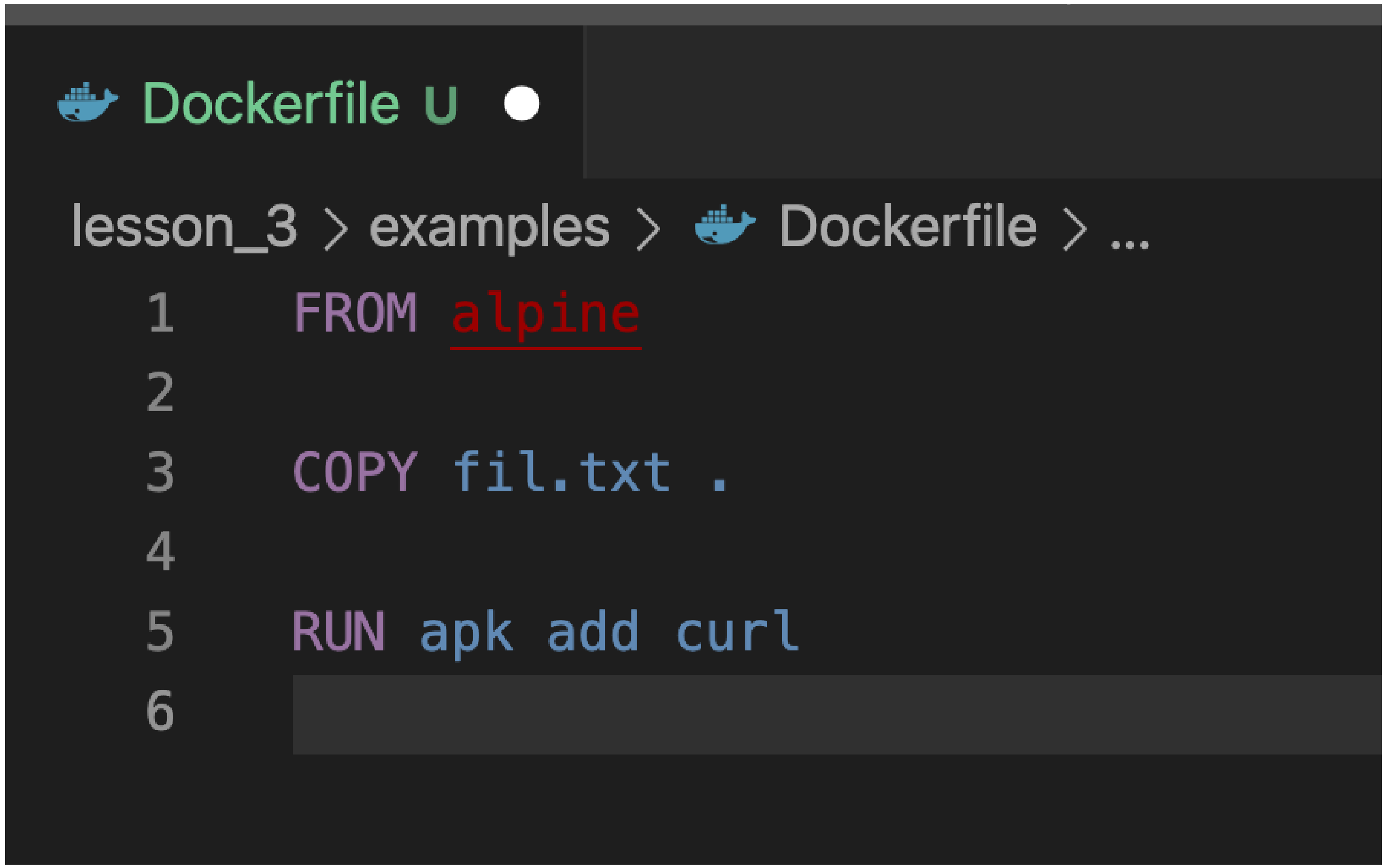
Gruppövning

- Testa att bygga en Dockerfile med:
    - FROM
    - RUN
    - ARG
    - Kommentar
    - Parserdirektiv

- Förslag:
    - Generera en fil med echo likt tidigare exempel
    - Testa att bygga från alpine, bullseye, bullseye-slim

NACKADEMIN

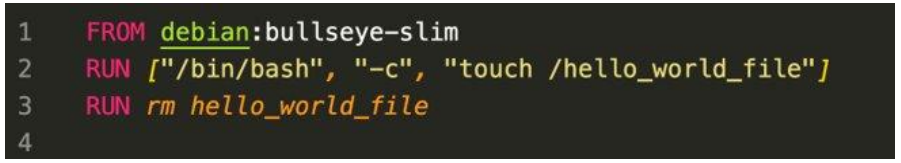
RUN



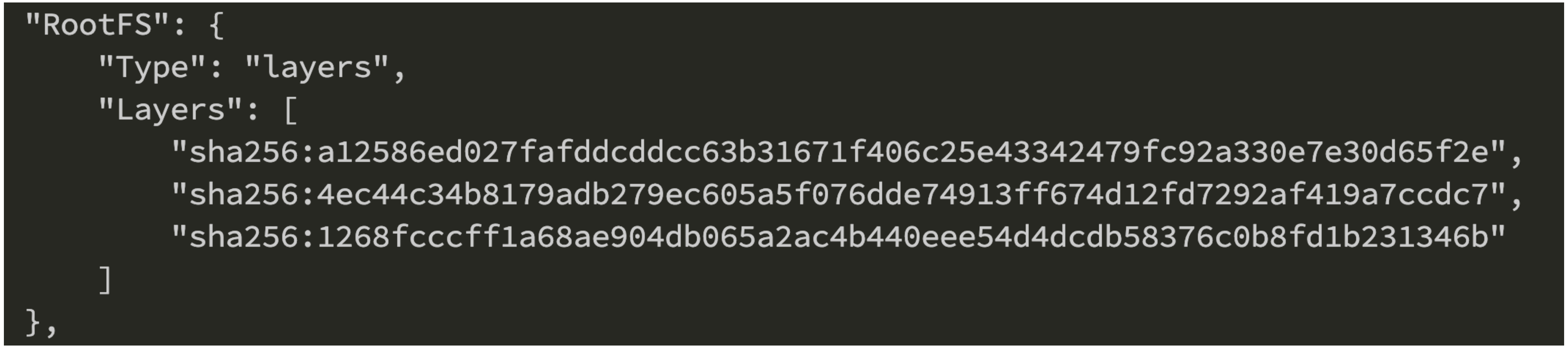
```dockerfile
FROM alpine

COPY fil.txt .

RUN apk add curl

```

NACKADEMIN

RUN

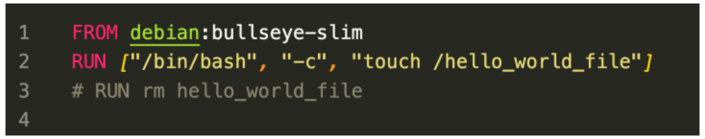


```dockerfile
FROM debian:bullseye-slim
RUN ["/bin/bash", "-c", "touch /hello_world_file"]
RUN rm hello_world_file

```

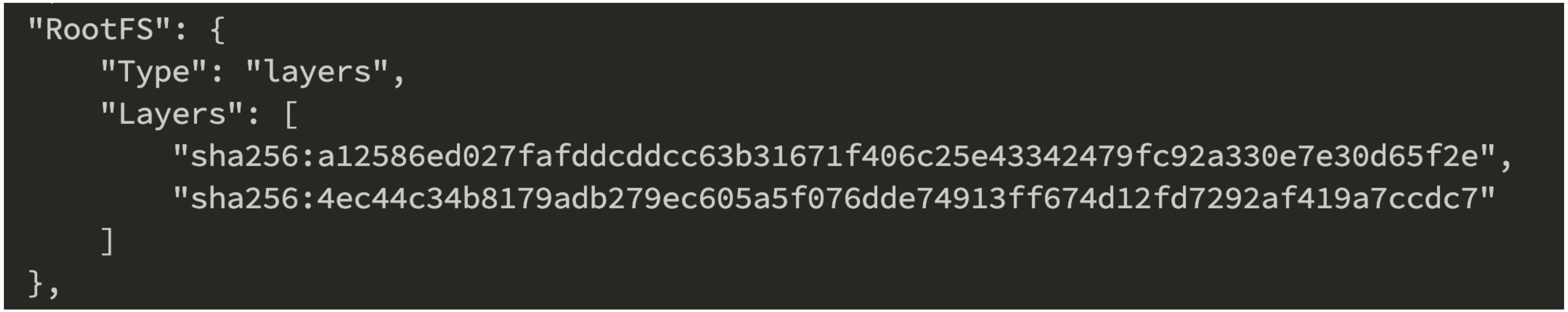


```
"RootFS": {
    "Type": "layers",
    "Layers": [
        "sha256:a12586ed027fafddcddcc63b31671f406c25e43342479fc92a330e7e30d65f2e",
        "sha256:4ec44c34b8179adb279ec605a5f076dde74913ff674d12fd7292af419a7ccdc7",
        "sha256:1268fcccff1a68ae904db065a2ac4b440eee54d4dcdb58376c0b8fd1b231346b"
    ]
},
```

RUN



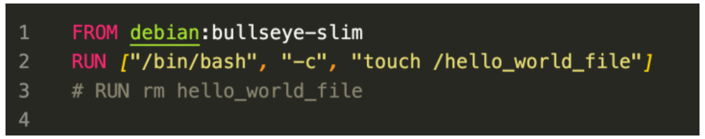
```dockerfile
FROM debian:bullseye-slim
RUN ["/bin/bash", "-c", "touch /hello_world_file"]
# RUN rm hello_world_file
```



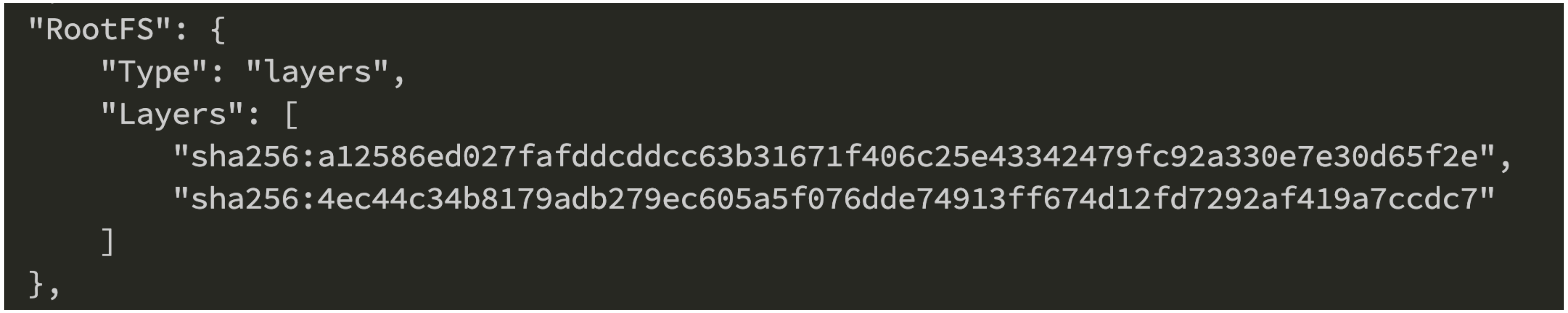
```json
"RootFS": {
    "Type": "layers",
    "Layers": [
        "sha256:a12586ed027fafddcddcc63b31671f406c25e43342479fc92a330e7e30d65f2e",
        "sha256:4ec44c34b8179adb279ec605a5f076dde74913ff674d12fd7292af419a7ccdc7"
    ]
},
```

NACKADEMIN

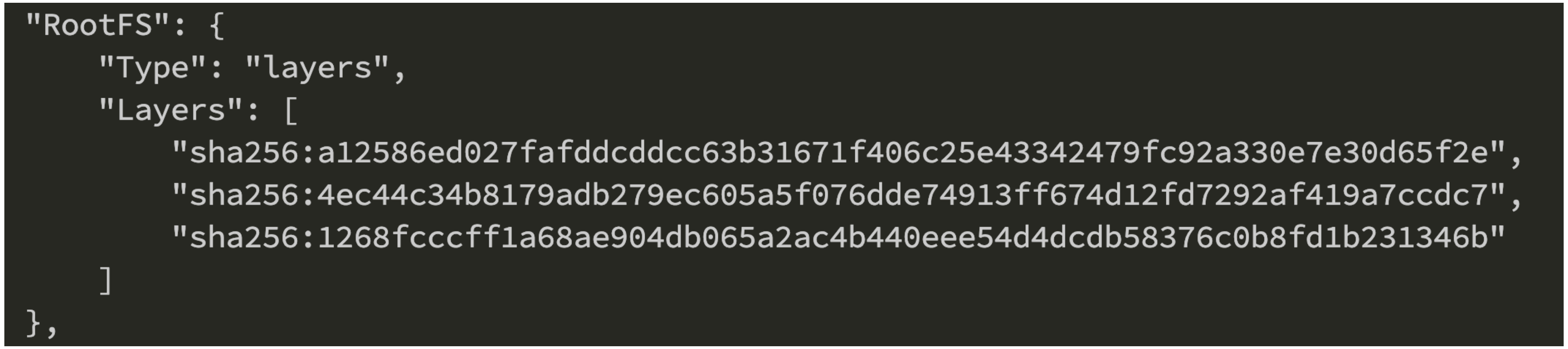
RUN



```
FROM debian:bullseye-slim
RUN ["/bin/bash", "-c", "touch /hello_world_file"]
# RUN rm hello_world_file

```



```
"RootFS": {
    "Type": "layers",
    "Layers": [
        "sha256:a12586ed027fafddcddcc63b31671f406c25e43342479fc92a330e7e30d65f2e",
        "sha256:4ec44c34b8179adb279ec605a5f076dde74913ff674d12fd7292af419a7ccdc7"
    ]
},
```



```
"RootFS": {
    "Type": "layers",
    "Layers": [
        "sha256:a12586ed027fafddcddcc63b31671f406c25e43342479fc92a330e7e30d65f2e",
        "sha256:4ec44c34b8179adb279ec605a5f076dde74913ff674d12fd7292af419a7ccdc7",
        "sha256:1268fcccff1a68ae904db065a2ac4b440eee54d4dcdb58376c0b8fd1b231346b"
    ]
},
```

Grupp Övning

Installera **curl** på en basimage exempelvis:

- alpine
- debian

Inspektera hur många lager du har:

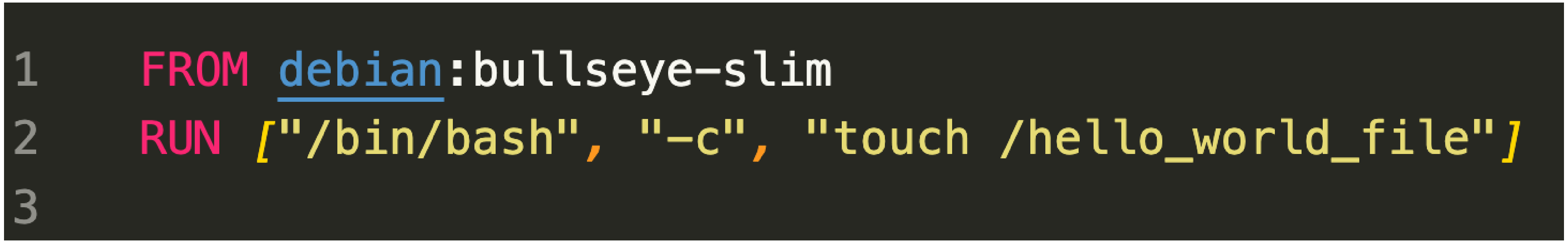
- Innan (ren image)
- Efter (med curl)

Vad händer om du samlar alla kommandon på en rad? Exempelvis i debian:

- Apt-get update && apt-get install -y curl

RUN

Du kan använda exec med andra shells



```
FROM debian:bullseye-slim
RUN ["/bin/bash", "-c", "touch /hello_world_file"]

```

NACKADEMIN

RUN tips & trix

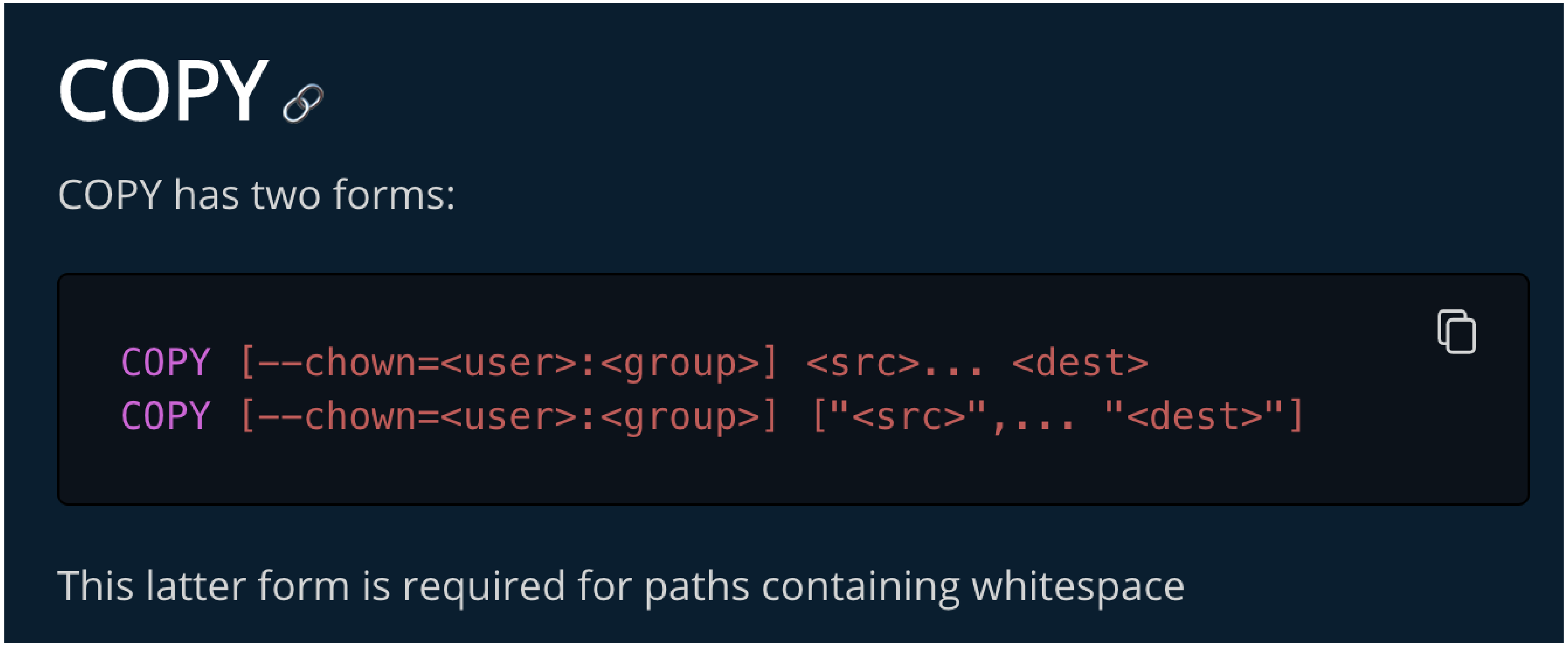
Extra läsning:

https://docs.docker.com/engine/reference/builder/#run---mount

https://docs.docker.com/engine/reference/builder/#run---network

NACKADEMIN

COPY



COPY

COPY has two forms:

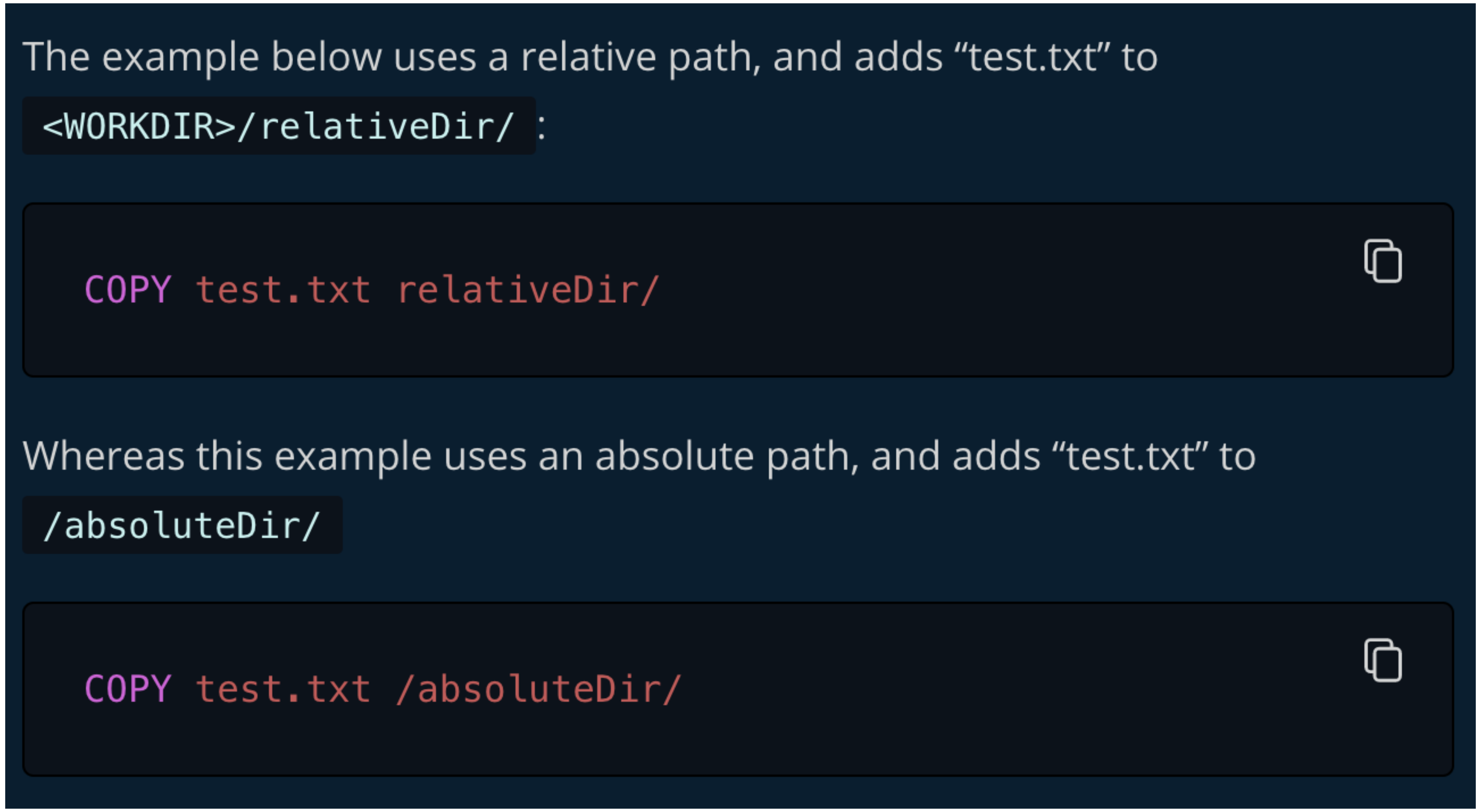
```
COPY [--chown=<user>:<group>] <src>... <dest>
COPY [--chown=<user>:<group>] ["<src>",... "<dest>"]
```

This latter form is required for paths containing whitespace

NACKADEMIN

Källa: https://docs.docker.com/engine/reference/builder/#copy

COPY



The example below uses a relative path, and adds "test.txt" to `<WORKDIR>/relativeDir/`:

```
COPY test.txt relativeDir/
```

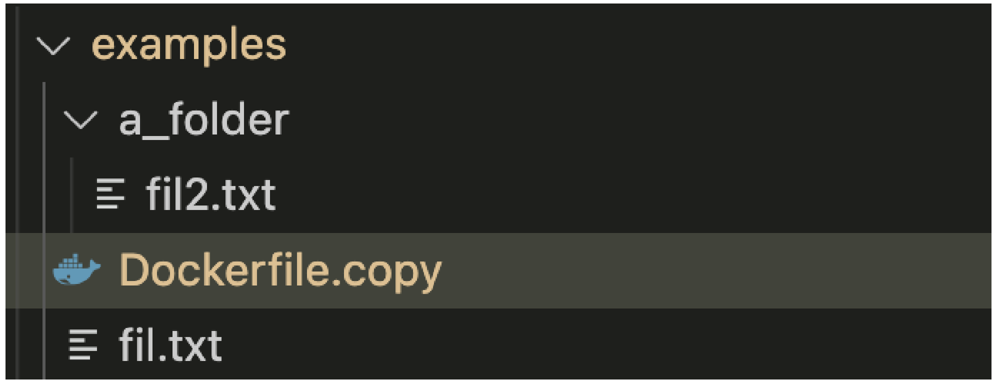
Whereas this example uses an absolute path, and adds "test.txt" to `/absoluteDir/`

```
COPY test.txt /absoluteDir/
```

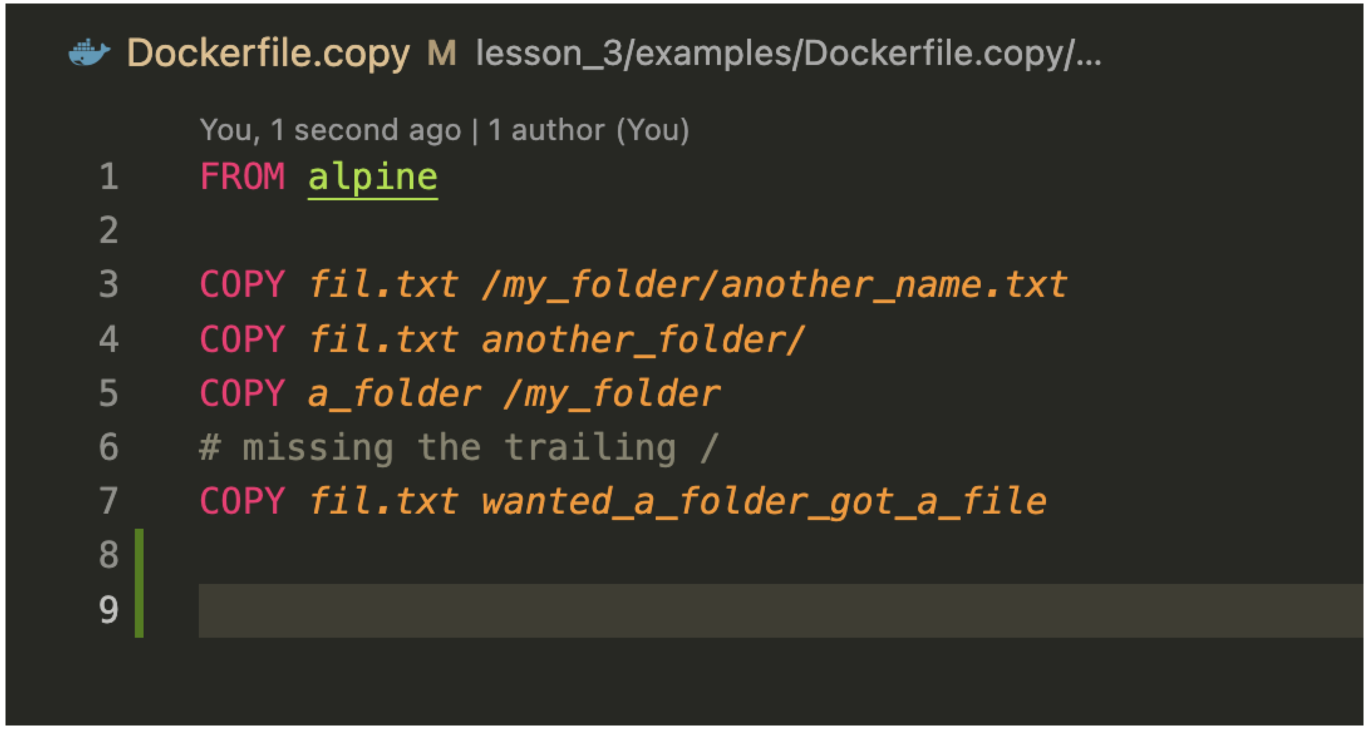
NACKADEMIN

Källa: https://docs.docker.com/engine/reference/builder/#copy

COPY



```
examples
  a_folder
    fil2.txt
  Dockerfile.copy
  fil.txt
```



Dockerfile.copy M lesson_3/examples/Dockerfile.copy/...

You, 1 second ago | 1 author (You)

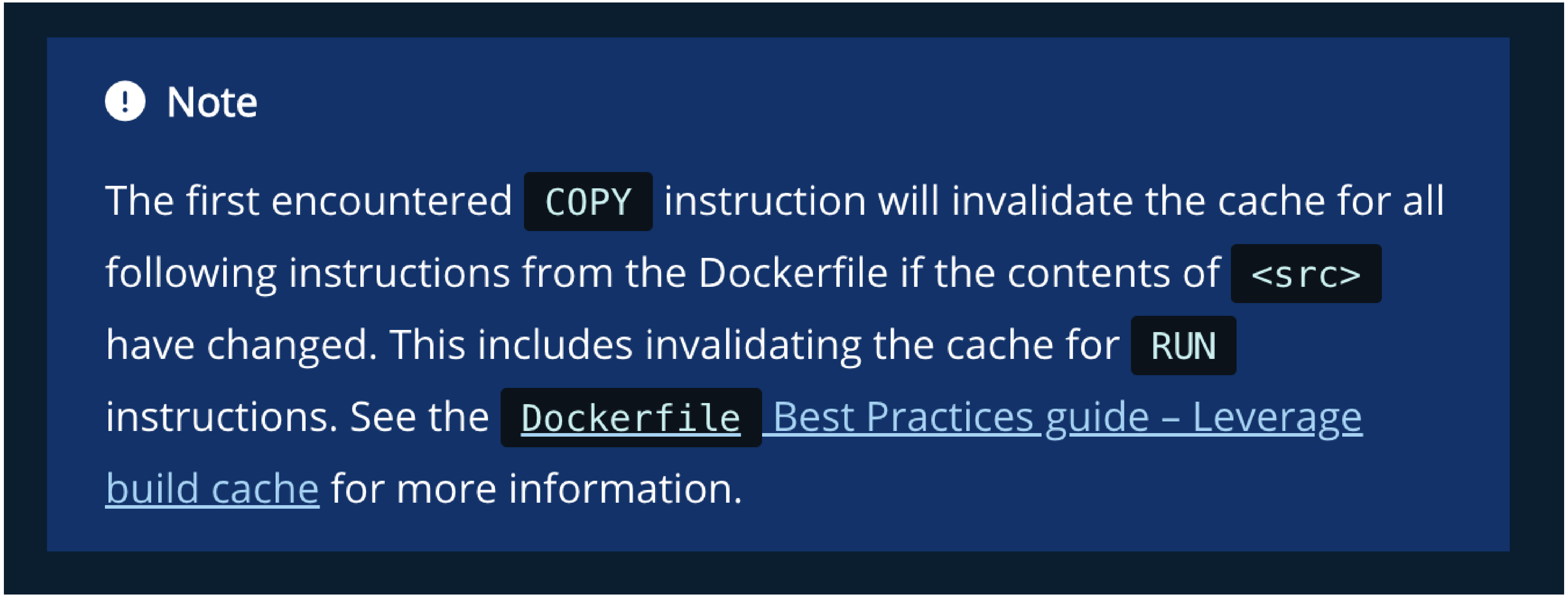
```dockerfile
FROM alpine

COPY fil.txt /my_folder/another_name.txt
COPY fil.txt another_folder/
COPY a_folder /my_folder
# missing the trailing /
COPY fil.txt wanted_a_folder_got_a_file


```

NACKADEMIN

COPY

- Vad blir resultatet? Bygg och testa!

COPY



Note

The first encountered `COPY` instruction will invalidate the cache for all following instructions from the Dockerfile if the contents of `<src>` have changed. This includes invalidating the cache for `RUN` instructions. See the `Dockerfile` Best Practices guide – Leverage build cache for more information.

NACKADEMIN

Källa: https://docs.docker.com/engine/reference/builder/#copy

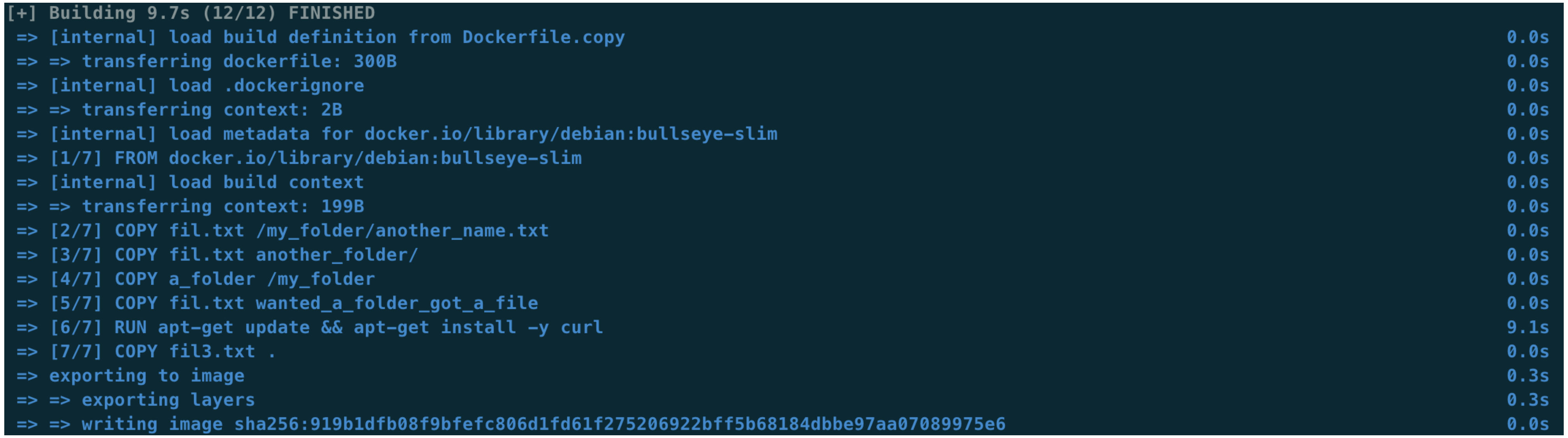
Gruppövning - Cache

- Stäng av eventuella körande containers
- Kör en docker system prune –all
- Se filer i foldern lesson_3/examples från devops21_contin repot
- Du kan köra build med flaggan –f Dockerfile.copy .
- Spara utskriften så ni kan jämföra
- Bygg igen! Vad händer?
- Testa att ändra fil1, fil2, och fil3 hur påverkar det?

NACKADEMIN

Källa: https://docs.docker.com/engine/reference/builder/#copy

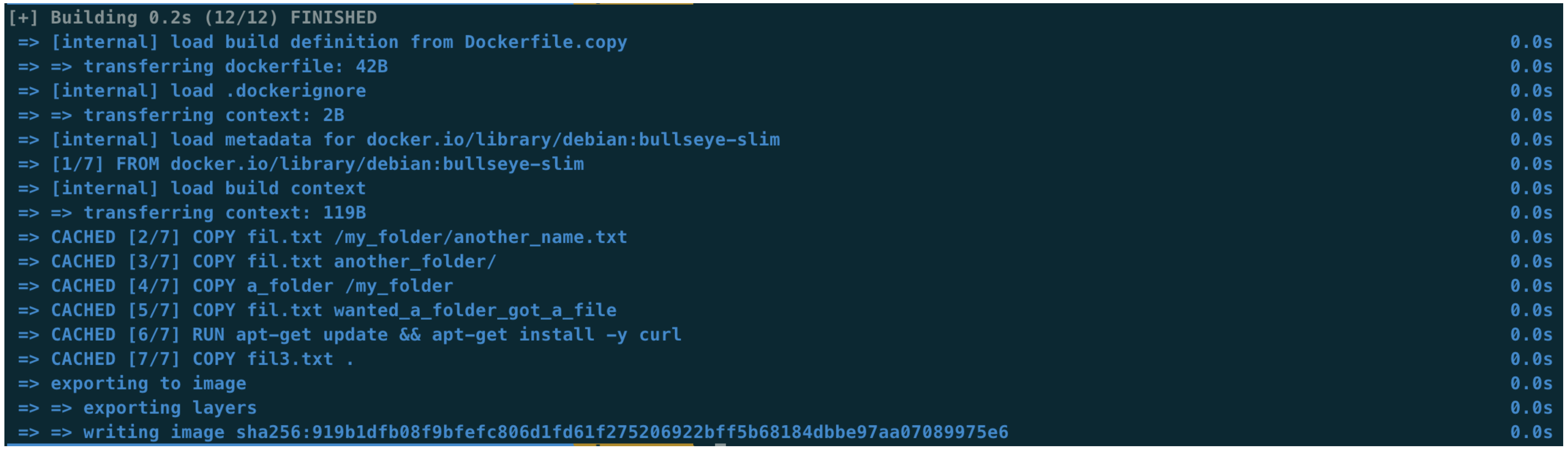
Gruppövning - Cache



```
[+] Building 9.7s (12/12) FINISHED
 => [internal] load build definition from Dockerfile.copy                                                     0.0s
 => => transferring dockerfile: 300B                                                                          0.0s
 => [internal] load .dockerignore                                                                             0.0s
 => => transferring context: 2B                                                                               0.0s
 => [internal] load metadata for docker.io/library/debian:bullseye-slim                                       0.0s
 => [1/7] FROM docker.io/library/debian:bullseye-slim                                                         0.0s
 => [internal] load build context                                                                             0.0s
 => => transferring context: 199B                                                                             0.0s
 => [2/7] COPY fil.txt /my_folder/another_name.txt                                                            0.0s
 => [3/7] COPY fil.txt another_folder/                                                                        0.0s
 => [4/7] COPY a_folder /my_folder                                                                            0.0s
 => [5/7] COPY fil.txt wanted_a_folder_got_a_file                                                             0.0s
 => [6/7] RUN apt-get update && apt-get install -y curl                                                       9.1s
 => [7/7] COPY fil3.txt .                                                                                     0.0s
 => exporting to image                                                                                        0.3s
 => => exporting layers                                                                                       0.3s
 => => writing image sha256:919b1dfb08f9bfefc806d1fd61f275206922bff5b68184dbbe97aa07089975e6                 0.0s
```

NACKADEMIN

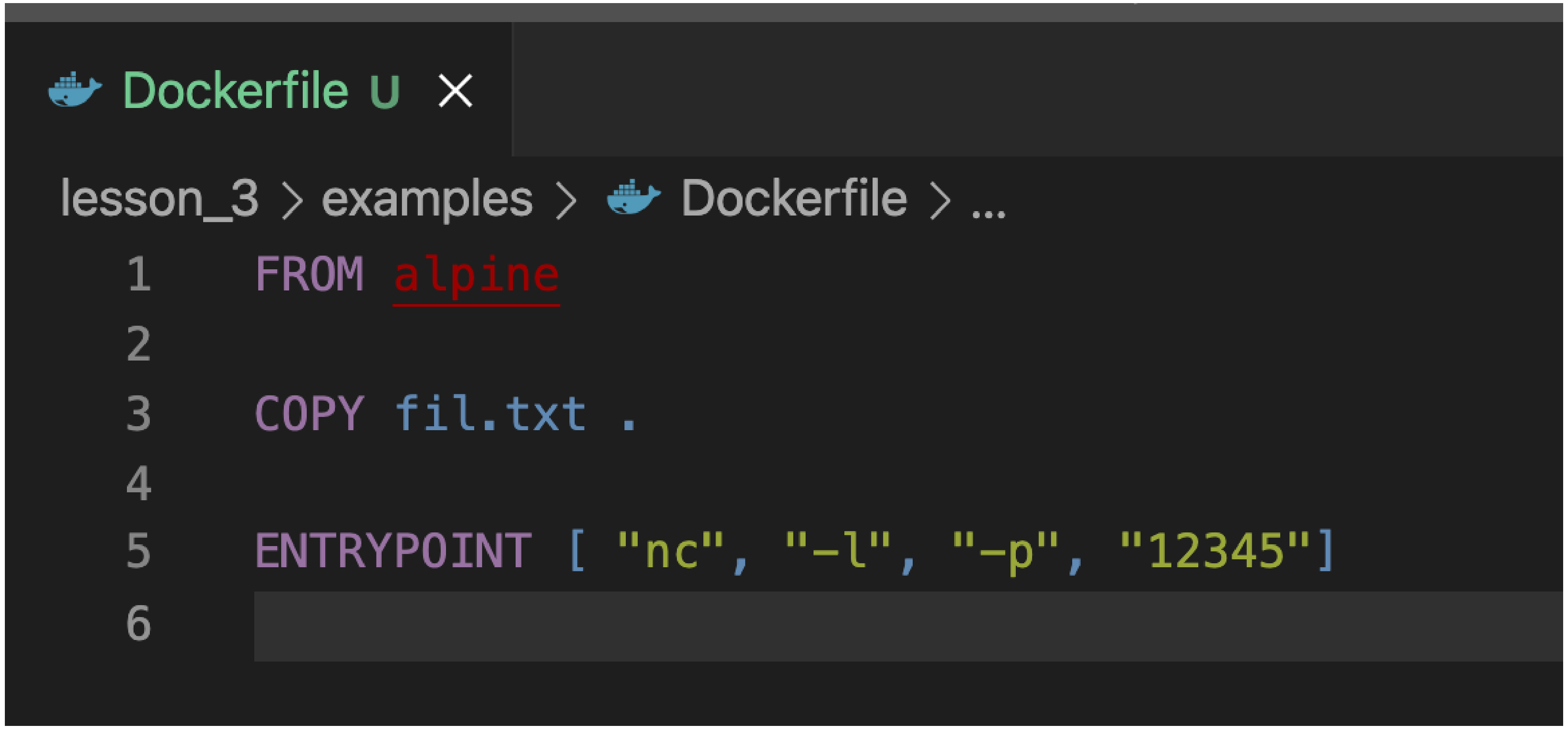
Gruppövning - Cache



```
[+] Building 0.2s (12/12) FINISHED
 => [internal] load build definition from Dockerfile.copy                                                  0.0s
 => => transferring dockerfile: 42B                                                                        0.0s
 => [internal] load .dockerignore                                                                          0.0s
 => => transferring context: 2B                                                                            0.0s
 => [internal] load metadata for docker.io/library/debian:bullseye-slim                                    0.0s
 => [1/7] FROM docker.io/library/debian:bullseye-slim                                                      0.0s
 => [internal] load build context                                                                          0.0s
 => => transferring context: 119B                                                                          0.0s
 => CACHED [2/7] COPY fil.txt /my_folder/another_name.txt                                                  0.0s
 => CACHED [3/7] COPY fil.txt another_folder/                                                              0.0s
 => CACHED [4/7] COPY a_folder /my_folder                                                                  0.0s
 => CACHED [5/7] COPY fil.txt wanted_a_folder_got_a_file                                                   0.0s
 => CACHED [6/7] RUN apt-get update && apt-get install -y curl                                             0.0s
 => CACHED [7/7] COPY fil3.txt .                                                                           0.0s
 => exporting to image                                                                                     0.0s
 => => exporting layers                                                                                    0.0s
 => => writing image sha256:919b1dfb08f9bfefc806d1fd61f275206922bff5b68184dbbe97aa07089975e6               0.0s
```

NACKADEMIN

ENTRYPOINT



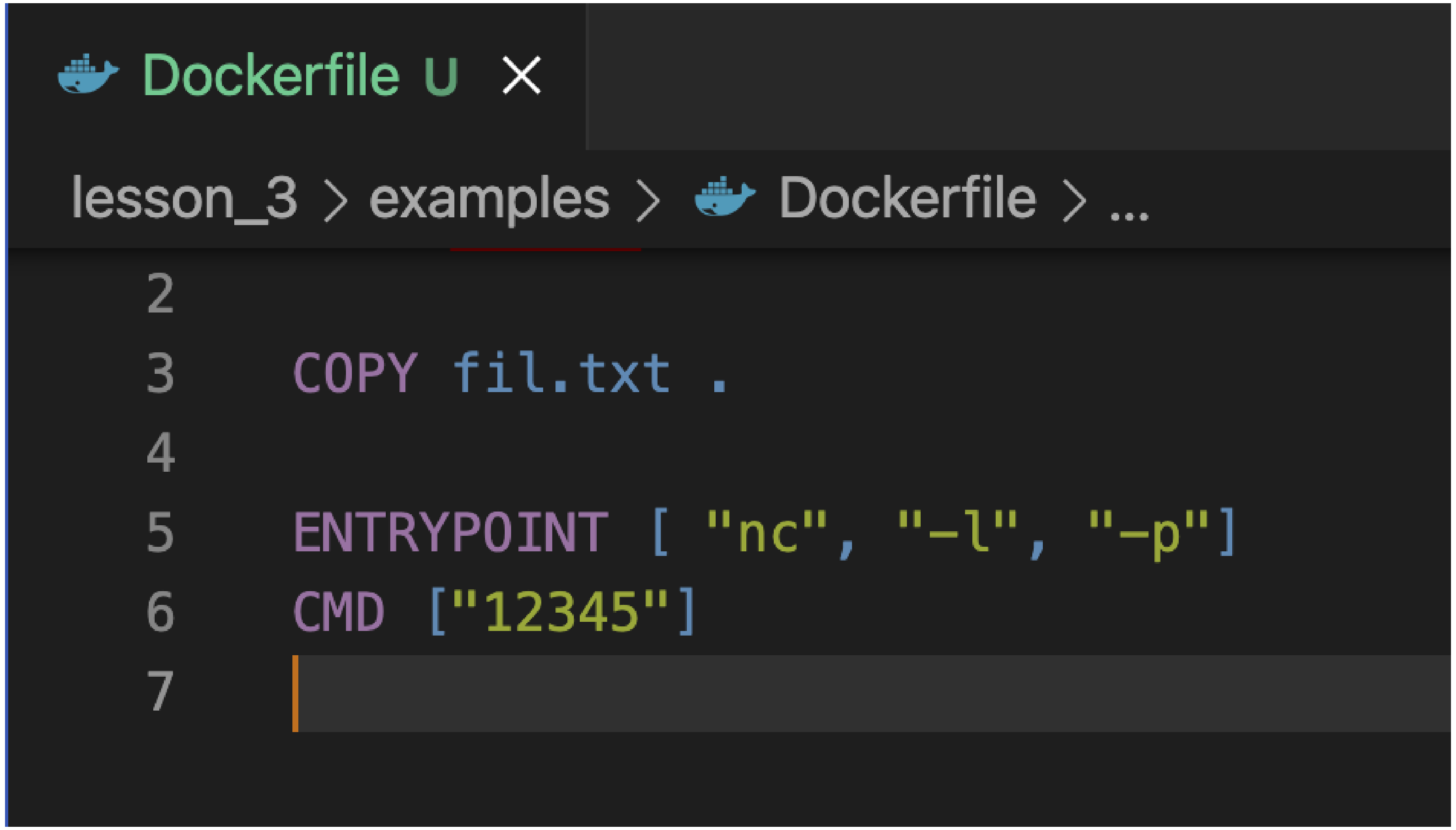
Dockerfile U ×

lesson_3 > examples > Dockerfile > ...

```
FROM alpine

COPY fil.txt .

ENTRYPOINT [ "nc", "-l", "-p", "12345"]

```

NACKADEMIN

CMD

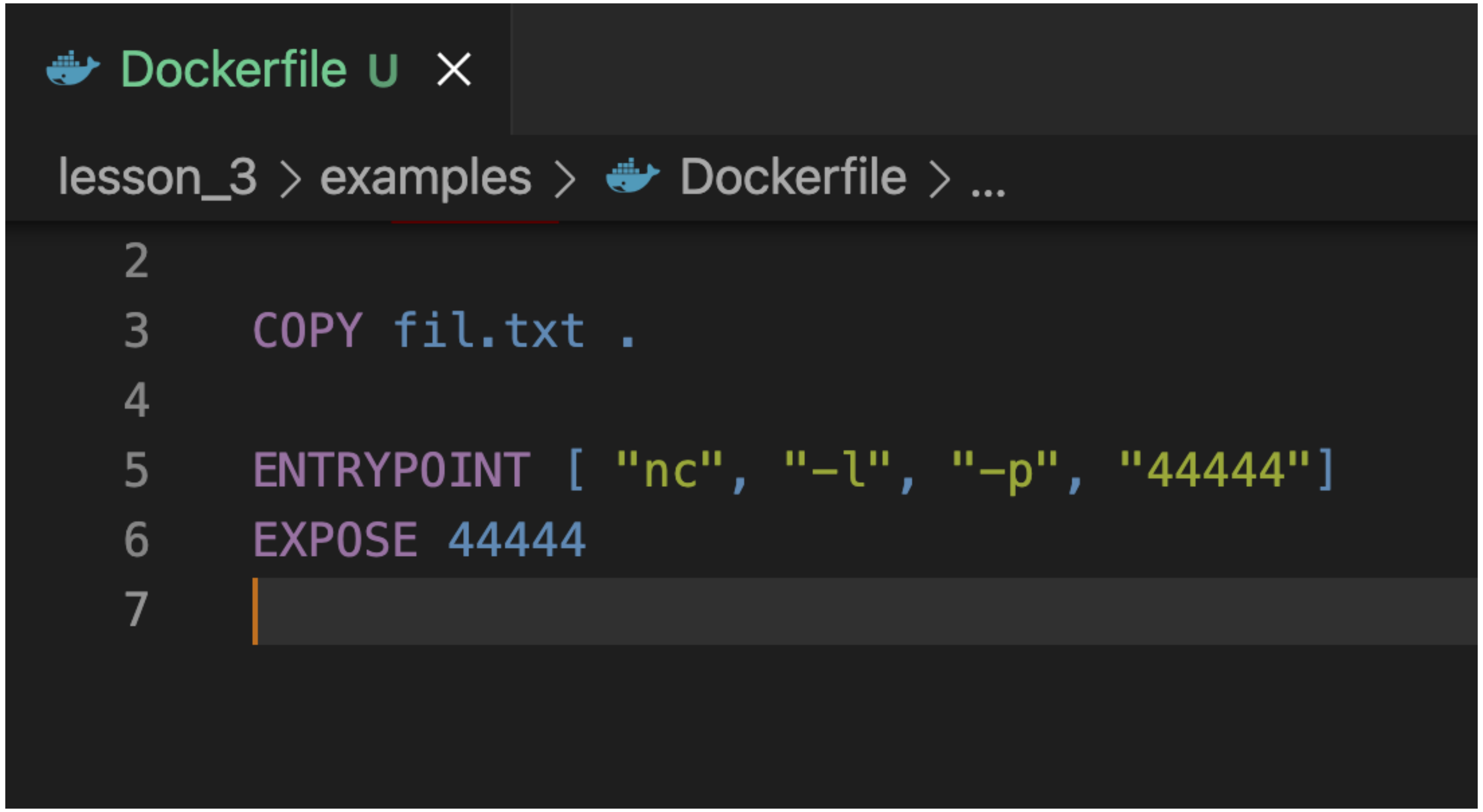

Dockerfile U

lesson_3 > examples > Dockerfile > ...

```

COPY fil.txt .

ENTRYPOINT [ "nc", "-l", "-p"]
CMD ["12345"]

```

NACKADEMIN

EXPOSE



```
COPY fil.txt .

ENTRYPOINT [ "nc", "-l", "-p", "44444"]
EXPOSE 44444
```

NACKADEMIN

Dockerfile exempel

Vi kollar på nginx:alpine

https://github.com/nginxinc/docker-nginx/blob/master/stable/alpine/Dockerfile

Vi kollar på python:3.11-slim

https://github.com/docker-library/python/blob/master/3.11/slim-bullseye/Dockerfile

https://hub.docker.com/_/python

NACKADEMIN

Övning

- Se studentportalen för GitHub Classroom länk

NACKADEMIN

Nästa lektion

- Minikube
- Kubernetes

NACKADEMIN