

## Image creation from a container

- created a container named lab2

```
mariemjrad@docker:~/lab2$ docker container run -it --name lab2 ubuntu bash
root@855bd0274bef:/# apt-get update
```

- installed figlet in lab2 container

```
Reading package lists... Done
root@855bd0274bef:/# apt install -y figlet
Reading package lists... Done
```

- testing figlet

```
root@855bd0274bef:/# figlet hello docker its lab2
```

[illegible]

- display containers

```
mariemjrad@docker:~/lab2$ docker container ls -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
855bd0274bef	ubuntu	"bash"	15 minutes ago	Exited (0) 19 seconds ago		lab2

- display the actual container with the changes applied over it
  - C : Changed
  - A : Added
  - D : Deleted

```
mariemjrad@docker:~/lab2$ docker diff lab2
C /usr
C /usr/share
C /usr/share/doc
A /usr/share/doc/figlet
A /usr/share/doc/figlet/changelog.Debian.gz
A /usr/share/doc/figlet/copyright
A /usr/share/doc/figlet/examples
A /usr/share/figlet
A /usr/share/figlet/8859-9.flc
A /usr/share/figlet/block.flf
A /usr/share/figlet/bubble.flf
```

- creating a new image with commit and provided the following tags:
  - -m : stands for message

- ```
docker container commit -m "added figlet" -a "Mariem Jrad" lab2 lab2:v2
552b130e005832594e87a39da3e0d5ce9600aeacc2b055
```

- ```
lab2@lab25-docker: /lab25-docker/images$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
lab2                 v2                 1272c338f9c9       6 minutes ago      125MB
lab2                 latest              1272c338f9c9       6 minutes ago      125MB
```

For this reason i created a new image from **lab2:v2** and named it **lab2:v3**

In the previous image creation I was providing the name. What if I create a new image without a name : **what would happen ? and is it possible to give it a name?**

==>it got actually created successfully but it did not give any significant information to recognize the image only Image ID

- ```
mariemjrad@docker:~/lab2$ docker image tag bb460c602fe5 ourfiglet
mariemjrad@docker:~/lab2$ docker images
```
- | REPOSITORY | TAG    | IMAGE ID     | CREATED       | SIZE  |
|------------|--------|--------------|---------------|-------|
| ourfiglet  | latest | bb460c602fe5 | 6 minutes ago | 125MB |

# HELLO

## Image creation using a Dockerfile

1. created an index.js that retrieves the hostname in this case it would be the container's name
2. Used alpine as the base OS image, added a Node.js runtime and then copied the source code in to the container. I specified also the default command to be run upon container creation.

```
mariemjrad@docker:~/docker-sample/lab2$ cat index.js
var os = require("os");
var hostname = os.hostname();
console.log("hello from " + hostname);

mariemjrad@docker:~/docker-sample/lab2$ cat Dockerfile
FROM alpine
RUN apk update && apk add nodejs
COPY . /app
WORKDIR /app
CMD ["node", "index.js"]
```

- build the image and named it using -t argument and tagged it "hello:v0.1"

```
CMD ["node", "index.js"]
mariemjrad@docker:~/docker-sample/lab2$ docker image build -t hello:v0.1 .
[+] Building 45.3s (9/9) FINISHED                                docker:default
=> [internal] load build definition from Dockerfile              0.5s
=> => transferring dockerfile: 131B                             0.2s
=> [internal] load metadata for docker.io/library/alpine:latest 0.0s
=> [internal] load .dockerignore                                0.4s
=> => transferring context: 2B                                    0.1s
=> [1/4] FROM docker.io/library/alpine:latest                  0.1s
=> [internal] load build context                                0.1s
=> => transferring context: 262B                                  0.0s
=> [2/4] RUN apk update && apk add nodejs                       42.0s
=> [3/4] COPY . /app   0.3s
=> [4/4] WORKDIR /app   0.1s
=> exporting to image   0.7s
=> => exporting layers   0.6s
=> => writing image sha256:8c9e1dab4d916bd66126ed83c6075883673ff285352a9 0.0s
=> => naming to docker.io/library/hello:v0.1                   0.0s
mariemjrad@docker:~/docker-sample/lab2$
```

```
mariemjrad@docker:~/docker-sample/lab2$ docker images
REPOSITORY          TAG             IMAGE ID          CREATED           SIZE
hello               v0.1           8c9e1dab4d91     35 seconds ago   74.8MB
ourfioret           latest         bb460c602fe5     3 hours ago      125MB
```

## Image layers

- The base image is **Alpine Linux**.
- **apk add nodejs** adds Node.js to the container.
- The last **CMD ["node", "index.js"]** means that when the container starts, it runs a Node.js application.

```
mariemjrad@docker:~/docker-sample/lab2$ docker image history hello:v0.1
IMAGE          CREATED          CREATED BY          SIZE      COMMENT
8c9e1dab4d91   2 hours ago     CMD ["node" "index.js"]  0B        buildkit.dockerfile.v0
<missing>      2 hours ago     WORKDIR /app        0B        buildkit.dockerfile.v0
<missing>      2 hours ago     COPY . /app # buildkit 188B      buildkit.dockerfile.v0
<missing>      2 hours ago     RUN /bin/sh -c apk update && apk add nodejs ... 67MB      buildkit.dockerfile.v0
<missing>      5 weeks ago     CMD ["/bin/sh"]      0B        buildkit.dockerfile.v0
<missing>      5 weeks ago     ADD alpine-minirootfs-3.21.2-x86_64.tar.gz /... 7.83MB    buildkit.dockerfile.v0
mariemjrad@docker:~/docker-sample/lab2$
```

⇒ the list of intermediate container images that were built along the way to creating your final Node.js app image

- i added a message that it should be displayed once the container with the new version of the image because this line will be modified in the index file is run **echo "console.log(\"this is v0.2\");" >> index.js**

```
mariemjrad@docker:~/docker-sample/lab2$ echo "console.log(\"this is v0.2\");" >> index.js
mariemjrad@docker:~/docker-sample/lab2$ cat index.js
var os = require("os");
var hostname = os.hostname();
console.log("hello from " + hostname);

console.log("this is v0.2");
mariemjrad@docker:~/docker-sample/lab2$
```

- building a new image using the updated application

```
mariemjrad@docker:~/docker-sample/lab2$ docker image build -t hello:v0.2 .
[+] Building 3.9s (9/9) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 131B
=> [internal] load metadata for docker.io/library/alpine:latest
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/4] FROM docker.io/library/alpine:latest
=> [internal] load build context
=> => transferring context: 191B
=> [2/4] RUN apk update && apk add nodejs
=> [3/4] COPY . /app
=> [4/4] WORKDIR /app
=> exporting to image
=> => exporting layers
=> writing image sha256:0b71d3342b9c177b7cfe8ad4ec29cf34cf07e6007ca8ef12476f8798a6caa45f
=> naming to docker.io/library/hello:v0.2
```

## Image Inspection

- **docker image inspect alpine** : the layers the image is composed of the driver used to store the layers the architecture / OS it has been created for/ metadata of the image

```
marlemjrad@docker:~/docker-sample/lab2$ docker image inspect alpine
[
  {
    "Id": "sha256:aded1e1a5b3705116fa0a92ba074a5e0b0031647d9c315983ccba2ee5428ec8b",
    "RepoTags": [
      "alpine:latest"
    ],
    "RepoDigests": [
      "alpine@sha256:a8560b36e8b8210634f77d9f7f9efd7ffa463e380b75e2e74aff4511df3ef88c"
    ],
    "Parent": ""
  }
]
```

- applying some filtering to get specific info about the image and because the inspect output is a json format it would be easy

```
marlemjrad@docker:~/docker-sample/lab2$ docker image inspect --format "{{ json .RootFS.Layers }}" alpine
["sha256:08000c18d16dadf9553d747a58cf44023423a9ab010aab96cf263d2216b8b350"]
```

⇒ Alpine is just a small base OS image so there's just one layer

- Doing the same thing but over the image i've been creating and modifying

```
marlemjrad@docker:~/docker-sample/lab2$ docker image inspect --format "{{ json .RootFS.Layers }}" hello:v0.1
["sha256:a0904247e36a7726c03c71ee48f3e64462021c88dafeb13f37fdaf613b27f11c", "sha256:decf6824c2c0cf51445b74ba71931cdd6bb43eff24af8f2df0515a403ff1221d", "sha256:1e3febfbe41b6db109f8e7d1bedfb50cd6991ab8f34c2c2eae048569b674e124", "sha256:5f70bf18a086007016e948b04aed3b82103a36bea41755b6cddfaf10ace3c6ef"]
marlemjrad@docker:~/docker-sample/lab2$ docker image inspect --format "{{ json .RootFS.Layers }}" hello:v0.2
["sha256:a0904247e36a7726c03c71ee48f3e64462021c88dafeb13f37fdaf613b27f11c", "sha256:decf6824c2c0cf51445b74ba71931cdd6bb43eff24af8f2df0515a403ff1221d", "sha256:a098196d87a59adf63081e84edba3fc85c4d03709090094aeb6892c530ac1321", "sha256:5f70bf18a086007016e948b04aed3b82103a36bea41755b6cddfaf10ace3c6ef"]
marlemjrad@docker:~/docker-sample/lab2$
```

- hello:v0.1 ⇒
  - This image consists of **4 layers**, each identified by its unique SHA-256 hash.
  - Each layer represents a change made during the Docker build process (adding files, installing software, setting environment variables).
- hello:v0.2 ⇒
  - This image also has **4 layers**, but one layer is different from **hello:v0.1**.
  - The third layer in **hello:v0.1**:  
**sha256:1e3febfbe41b6db109f8e7d1bedfb50cd6991ab8f34c2c2eae048569b674e124**  
 was replaced by:  
**sha256:a098196d87a59adf63081e84edba3fc85c4d03709090094aeb6892c530ac1321** in hello:v0.2.
  -