Error:

is the docker daemon running?

Sudo systemctl start docker

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

برنامه ای که قرار است نصب شود در داکر به عنوان image شناخته میشود و محیطی که این برنامه ها قرار است در آن ها اجرا شوند به عنوان container شناخته میشوند(توجه شود که نصب iamge در خود docker میباشد)

===================================================================================

دستور sudo su در ترمینال لینوکس باعث میشود شما به عنوان کاربر root شناخته شوید

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

docker [option] COMMAND حالت کلی دستورات docker میباشد

docker help COMMAND

docker pull hello-world:latest

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

docker ps ====>نمایش containerهایی که بسته نشده اند و هنوز فعال هستند

docker ps -a ====> نمایش containerهایی که بسته شده اند

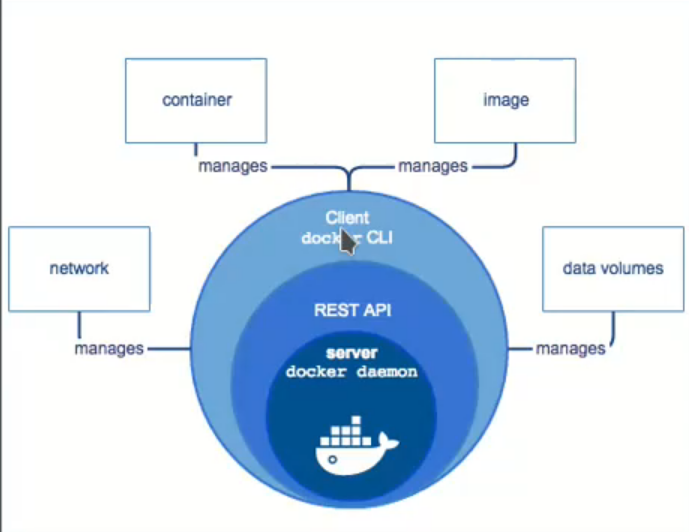
میتوان آپشن های هر دستور docker را باهم ترکیب کرد مثلا :

sudo docker ps -aq

===================================================================================

نکته مهمی که باید به آن توجه کرد این است که داکر برای اجرا یک image حتما یک container جدید میسازد حتی اگر آن image تکراری باشد و قبلا ران شده باشد به زبان دیگر اگر بخواهید ۳ بار image hello-world را اجرا کنید در پایان شما ۳ container خواهید داشت

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ



مفهوم کلی داکر در این عکس بیان شده است که ما در docker cli دستورات را وارد میکنیم این دستورات در قسمت rest api ترجمه میشوند و در قسمت

docker daemon اجرا میشود و به این ۳ بخش باهم docker engin گفته میشود

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

اختصاص نام برای container:

docker run –name first hello-world

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

حذف یک container:

docker rm name\_container\_archive

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

حذف تمام containerها:

docker container prune

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

هنگامی که ما از دستور run استفاده میکنیم اگر image در local ما وجود داشته باشد آن را اجرا میکند اما اگر وجود نداشته باشد ابتدا دانلود کرده سپس اجرا میکند

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

اجرا در یک container و حذف آن :

docker run –rm name\_\_image

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

همان طور که میدانید busybox یک لینوکس سبک میباشد اگر هنگام اجرا image

busybox از آپشن it استفاده کینم باعث میشود که وارد busybox بشویم

docker run -it –rm busybox

i (interactive):

stdin(standard input) را برای من نگه دار تا زمانی که لازم داشته باشم منظور از stdin جایی است که میتوانم از طریق آن دستورات را وارد کنم(کیبورد)

t (tty) :

کل سیستم شما را به یک محیطی میبرد که بتوانید با آن کار کنید

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

برای خروج میبایست exit را تایپ کنیم و یا ctrl+D را بزنیم

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

دستور rmi باعث حذف یک یا چند image میشود

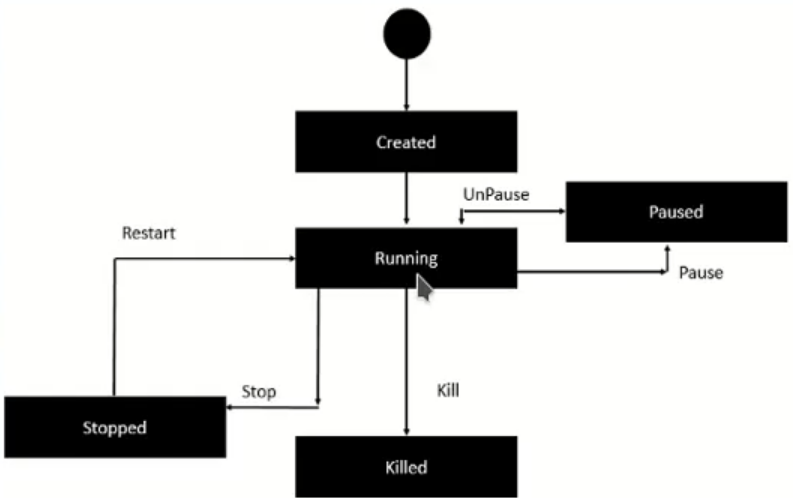
===================================================================================

بعضی وقت ها لازم است که container ما تا زمانی که ما بخواهیم در حال اجرا باشد برای این منظور هنگام run از آپشن detach استفاده میکنیم که container را در background اجرا میکند و در حالت اجرا باقی میگذارد

docker run [-d or –detach ] redis

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

container life cycle:



نکته ای کع در اینجا باید دقت شود این است که اگر containerی با دستور kill هم متوقف شده باشد هم میتوان دوباره با دستورات start و restart آنرا فعال کرد

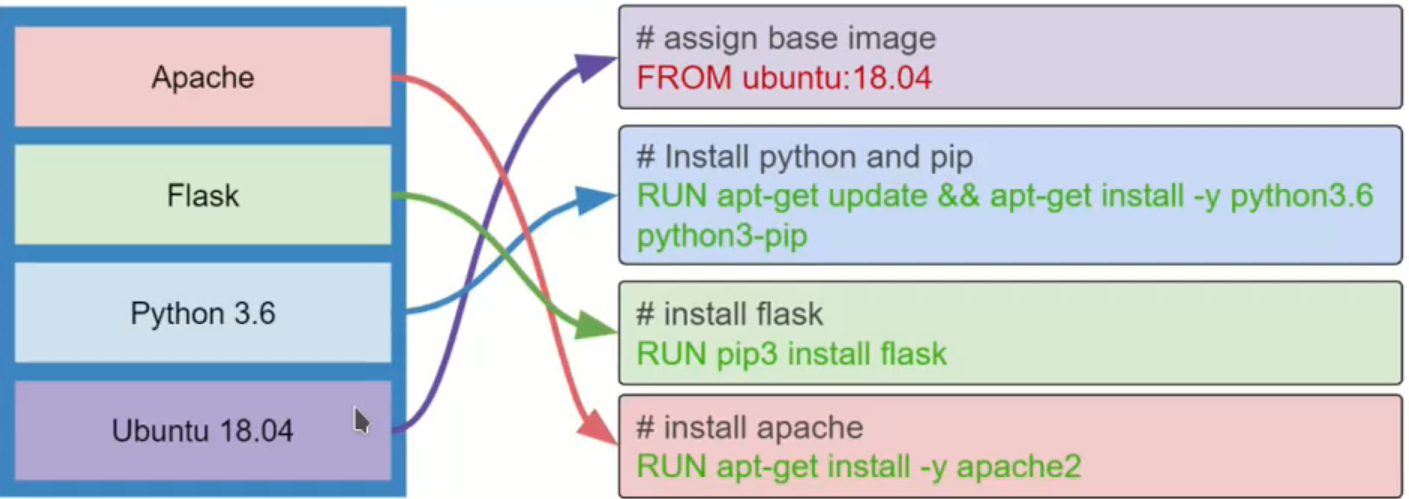
==========================================================

با دستور exec میتوانیم یک دستور را به یک container در حال اجرا ارسال کنیم و خروجی دریافت کنیم:

docker exec name\_container name\_command

docker exec -it rd bath

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ



===================================================================================

Docker می تواند با خواندن دستورالعمل های Dockerfile به طور خودکار imageها را بسازد. Dockerfile یک سند متنی است که شامل تمام دستوراتی است که کاربر می تواند در خط فرمان برای جمع آوری یک image فراخوانی کند. با دستور

docker build کاربران می توانند یک برنامه قابل اجرا خودکار ایجاد کنند که چندین دستورالعمل خط فرمان را به طور متوالی اجرا کند.

docker build -t name:tag address

. docker buld -t req:1.10

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

در فایل Dockerfile نکته ی مهمی که وجود دار WORKDIR میباشد

WORKDIR /sajjad:

این خط کد به این معنی میباشد که در دایرکتوری که هستی پوشه sajjad اگه وجود داشت برو داخلش و محل اصلی از این به بعد این پوشه میباشد و اگر وجود نداشت بساز و داخلش برو و به عنوان پوشه اصلی در نظر بگیر

===================================================================================

docker login این دستور برای ست کردن username و password شما در ترمینال میباشد

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

ساخت یک image جدید از image موجود :

docker tag old\_name new\_name

docker tage req:1.10 ahmadalim/req:one

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

آپلود کرد در dockerhub:

docker push ahmadalim/req:one

===================================================================================

Port Forwarding:

این کار به صورت‌های مختلفی انجام می‌شود و از آپشن p- استفاده می‌شود :

8080:80 ===> در اینجا این منظور است که پورت 8080 لکال من را به پورت 80 کانتینر متصل کن

توجه داشته باشید که میتوانید host خود را مشخص کنید اگر host خود را مشخص نکنید به طور پیش‌فرض 0.0.0.0 درنظر گرفته می‌شود

192.168.1.100:8080:80

===================================================================================

load and save:

گاهی وقت‌ها لازم است که imageی را بصورت فایلی ذخیره کرده تا بتوانید از آن در جای دیگر استفاده کنید برای این منظور کافی از دستورات save و load استفاده کنید:

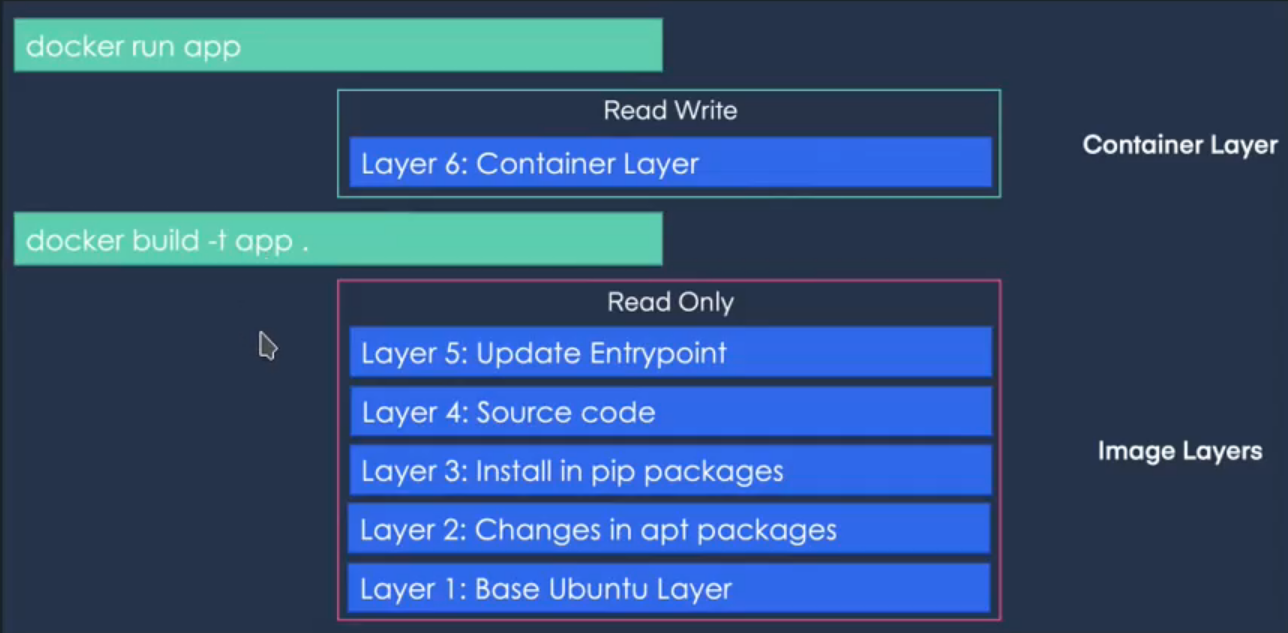
docker save -o alpine.tar alpine

آپشن o نام فایل خروجی را مشخص میکند و همیشه باید پسوندش tar باشد

docker load -i alpine.tar

آپشن I نام فایل ورودی را مشخص میکند

===================================================================================



commit کردن لایه‌های جدید یک container:

همان‌طور که از عکس بالا مشخص است لایه‌هایی که در image اصلی وجود دارند read only هستند و ما نمیتوانیم آن هارا تغییر دهیم اما میتوانیم لایه‌های دیگری از طریق container layer اضافه کنیم برای این منظور باید از image مورد نظر یک container ساخته شود و تغییراتی که مد نظرمان میباشد اعمال شود سپس با دستور زیر یک image جدید همراه با تغییرات ما ساخته می‌شود :

docker container commit name\_container name\_new\_image

docker container commit py python:first

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

با درستور history میتوانیم تاریخچه دستوراتی که در ترمینال زده ایم را ببینیم

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

قبل از اینکه یک container را commit کنیم میتوانیم با دستور diff ببینم چه تغییراتی در این container به وجود آمده است :

docker diff name\_container

docker diff py

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

هنگام commit کردن میتوانیم با آپشن های a و m نویسنده این تغییرات و توضیحی درباره این تغییرات مشخص کنیم :

docker container commit -a ‘sajjad’ -m ‘add test.txt’ py python:first

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

با استفاده از دستور inspect میتوانیم اطلاعاتی درباره ی یک image ببینیم :

docker inspect python:first

در اینجا نویسنده و متن پیغام مشخص می‌شود

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

با استفاده از دستور info میتوانیم اطلاعاتی درباره ی خود docker به دست آوریم:

docker info

===================================================================================

هنگام ساخت Dockerfile در اولین خط هنگامیکه میخواهیم base image را مشخص کنید میتوانیم از scratch استفاده شود که باعث می‌شود یک image خالی برای شما ساخته شود و خود شما باید تمام configها را مشخص کنید که پیشنهاد نمیشود چون بسیار مشکل میباشد:

FROM scratch

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

در Dockerfile دستور MAINTAINER وجود دارد که میتوانید اطلاعاتی درباره کسی که این Dockerfile را میسازد بنویسید:

MAINTAINER ‘ahmadalim mr.ahmadalimoshtaghi@gmail.com’s

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

در Dockerfile برای کامنت گذاری میتوانیم از علامت # استفاده کنیم

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

دستور RUN برای اجرا کردن هر دستوری داخل ترمینال استفاده می‌شود (در alpine با دستور apk میتوانیم برنامه‌ها رو نصب کنیم):

RUN apk update

Run apk add nano

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

دستور CMD هم ماننده RUN برای اجرا کردن استفاده می‌شود اما با این تفاوت‌ها که از دستور CMD برای اجرا کردن برنامه‌ها استفاده می‌شود (مثلاً برنامه‌های پایتونی) و از دستور RUN برای اجرا کردن دستورات سیستمی استفاده می‌شود تفاوت دوم این است که از دستور RUN به هر تعدادی میتوانیم داخل Dockerfile استفاده کنید اما از دستور CMD تنها یکبار میتوانید استفاده کنید و اگر به تعدادی از دستور CMD استفاده کرده باشید فقط آخرین CMD اجرا می‌شود

CMD python ./main.py

تفاوت سوم این است دستور RUN هنگام ساخت image صدا زده می‌شود اما دستور CMD هنگامیکه image اجرا می‌شود صدا زده می‌شود

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

هنگام ساخت image از Dockerfile به این صورت عمل می‌شود که هر دستور به عنوان یک لایه در نظر گرفته می‌شود و هنگام اجرای هر دستور یک image جدید ساخته می‌شود و به image های قبلی اضافه می‌شود پس به زبان دیگر هنگام بیلد Dockerfile به تعداد دستورات موجود در Dockerfile یک image ساخته می‌شود

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

اگر از قبل یک image ساخته باشید و اکنون یک image جدید با همان نام قبلی ایجاد کنید docker آن image قبلی را به عنوان dangling image در نظر میگیرد و اسم آن را None میگذارد

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

برای حذف dangling image میتوانیم از دستور rmi همراه با image id استفاده کنیم اما اگر dangling image هایمان زیاد بود میتوانیم از روش زیر استفاده کنیم:

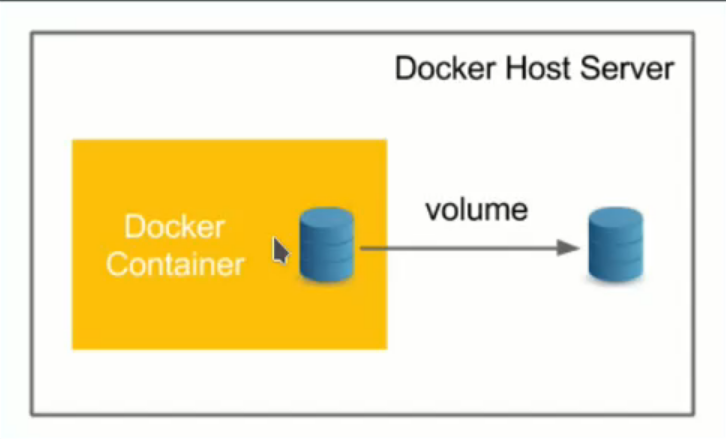
docker rmi $(docker images -q –filter “dangling=true”)s

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

دستور EXPOSE یک پورت را خالی میکند که بتوانیم به آن وصل شویم

===================================================================================

مفهوم volume ها در داکر به این معنی میباشد که ما اطلاعات مهمی را در یکcontainer داریم برای ذخیره این اطلاعات از volume ها استفاده می‌شود



دستورات volumeها:

docker volume create my\_volume

docker volume ls

docker volume inspect my\_volume

docker volume rm my\_volume

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

وصل کردن container به یک volume:

docker run -it -d –name ap -v v1:/home alpine

آپشن v در اینجا به این معنی است که volume که به این container متصل شده است v1 میباشد و در container هر چیزی که در دایرکتوری home ذخیره می‌شود را به volume v1 منتقل کن

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

هر اتفاقی که در container بیافتد در داخل volume وجود دارد و بر عکس این قضیه هم وجود دارد

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

محل ذخیره volume ها به صورت زیر میباشد:

s/var/lib/docker/volumes/v1/\_data

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

متصل کردن یک container به volume از یک container دیگر:

docker run -it -d –name ap2 –volumes-from ap alpine

در این مثال container ap2 به volumeی که برای container ap بود متصل شد

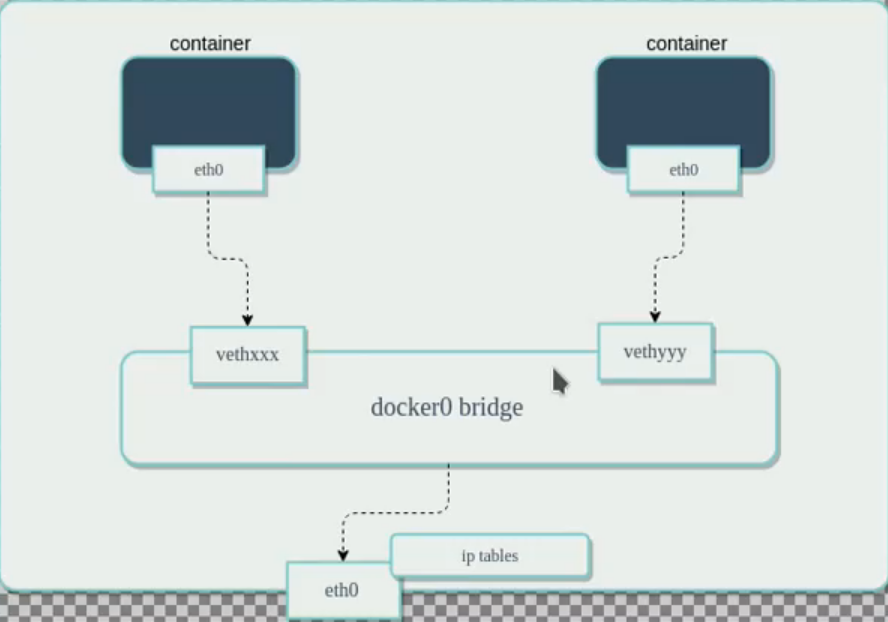
ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

حال میخواهیم که container ap3 فقط بتواند اطلاعات را بخواند و نتواند اطلاعات را تغییر دهد:

docker run -it -d –name ap3 –volumes-from ap:ro alpine

===================================================================================

Network:



دو نوع ارتباط بین container میتوان بوجود آورد یکی bridge و دیگری host

که بصورت پیش‌فرض bridge میباشد bridge برای زمانی مناسب است که containerهایتان در یک سیستم میباشند

ــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

دستورات network:

docker network create one

docker network inspect one

docker network connect one py

دقت داشته باشید که وقتی چندین container را به یک network متصل کردید برای آدرس دهی کافیست که فقط اسم آن container را صدا بزنید

===================================================================================

docker compose:

ابزاری است که میتوانیم چندین container را همزمان باهم در یک host اجرا کنیم به این صورت میباشد که شما یک file yaml ایجاد میکنیم و configهای خود را به آن اضافه میکنیم

اولین دستوری که باید نوشته شود version که باعث هماهنگ کننده compose و docker engin می‌شود