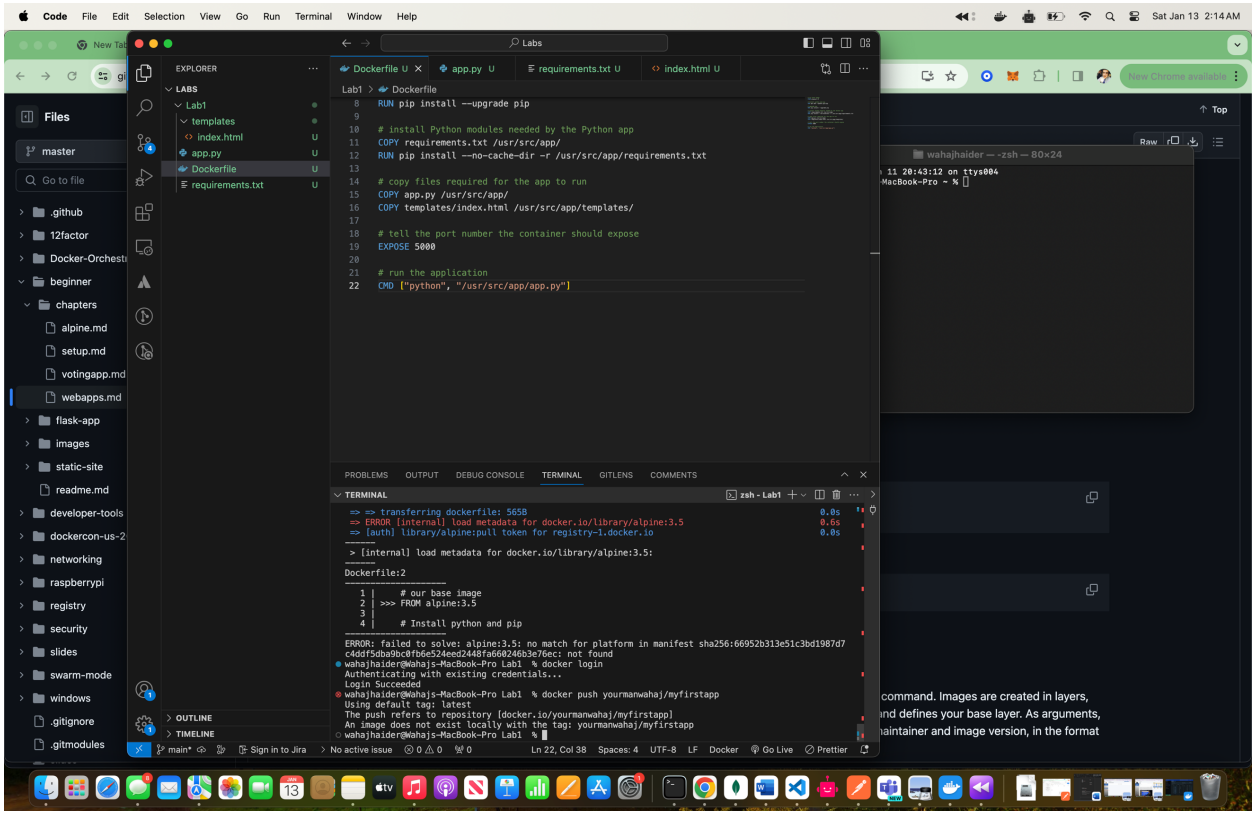
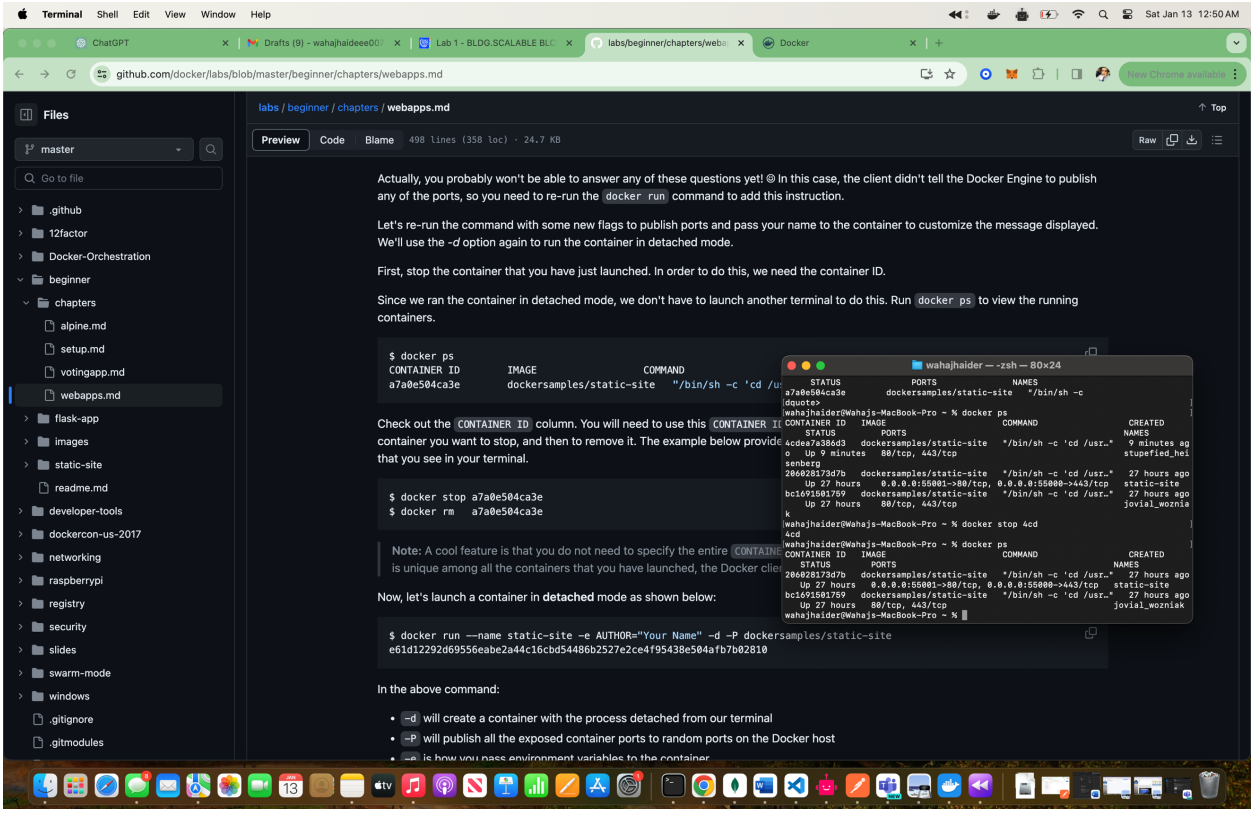
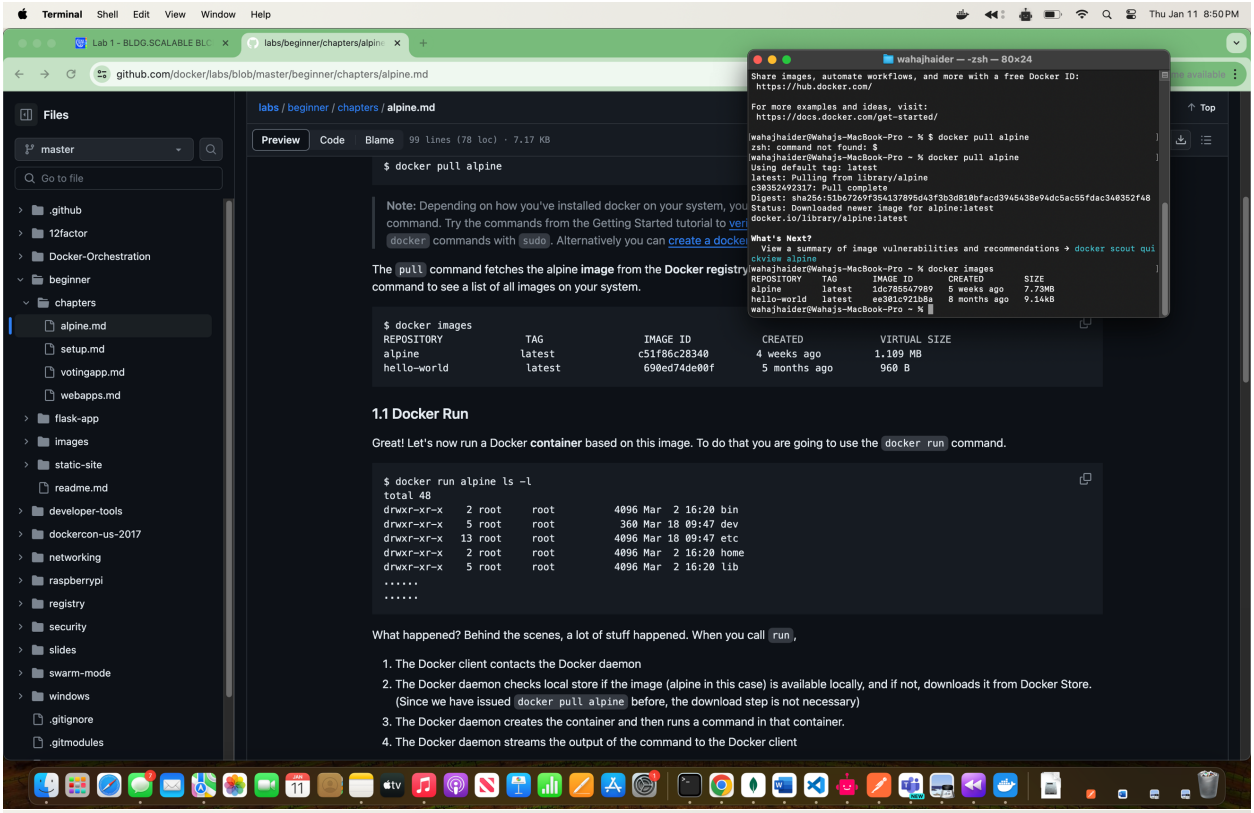
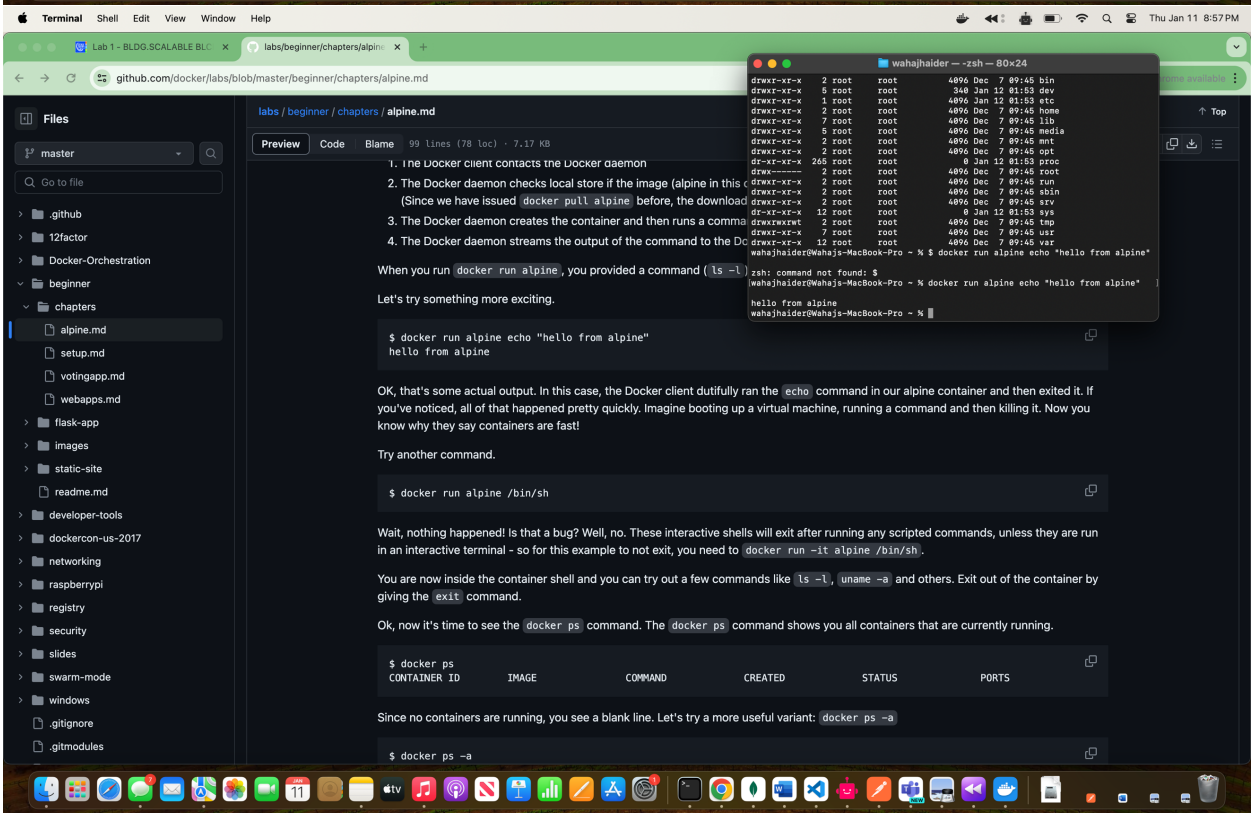
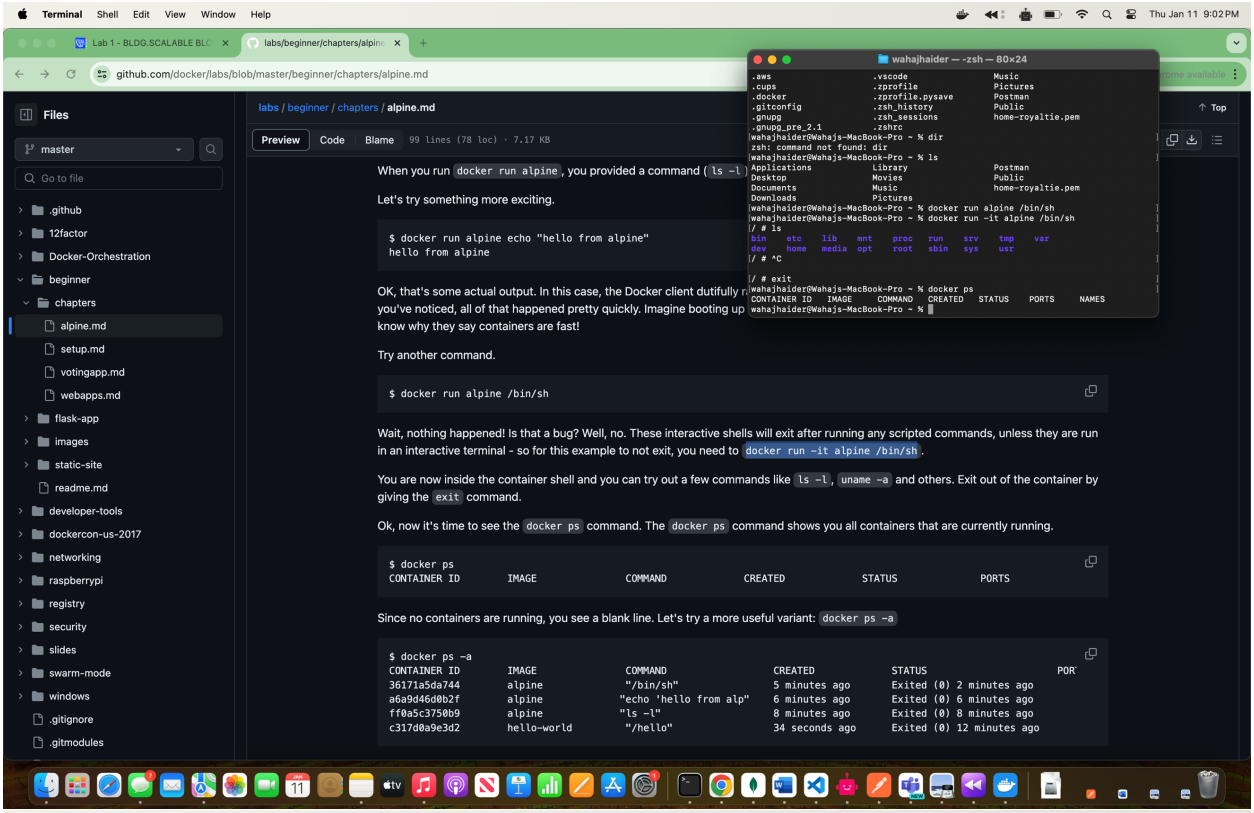
Error in repository to build image. Tried correcting it but then need to update app.py to support python 2 and it still not working.

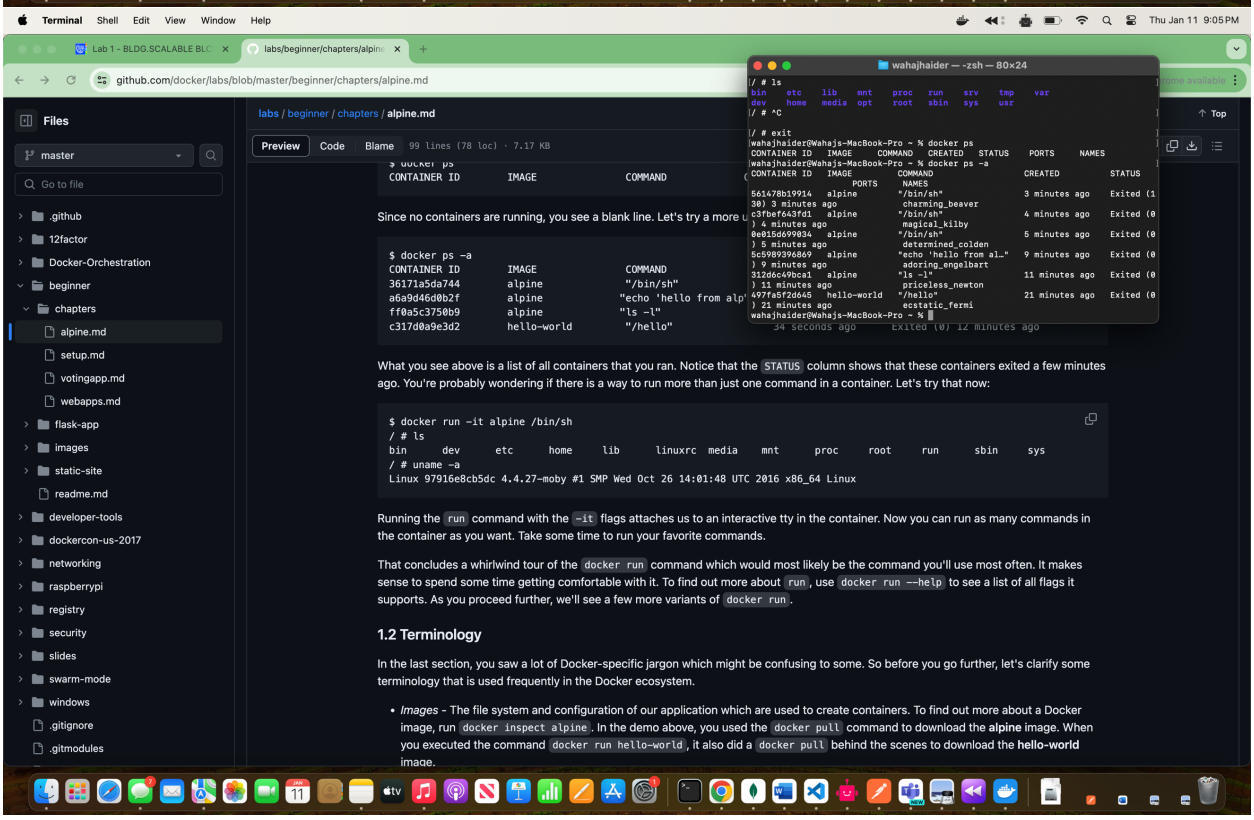


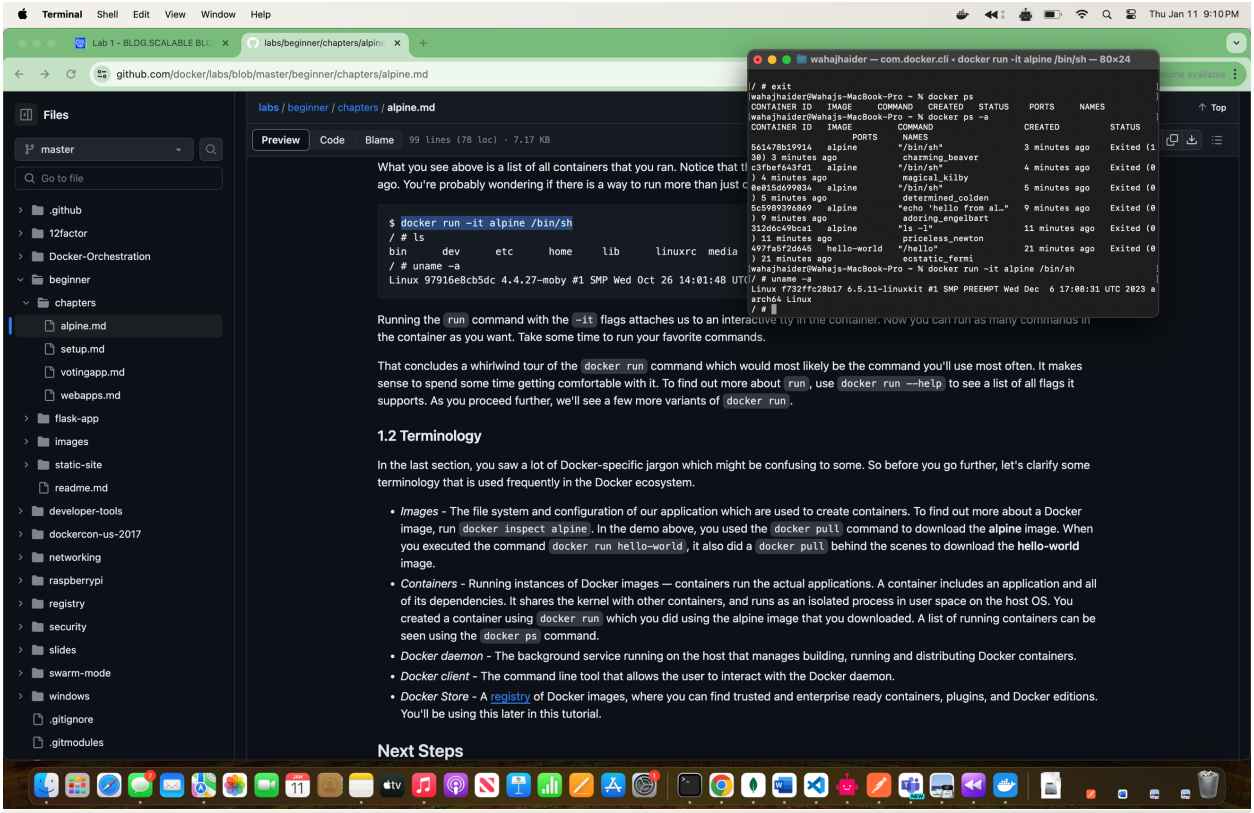

Terminal Shell Edit View Window Help
Sat Jan 13 12:50 AM
github.com/docker/labs/blob/master/beginner/chapters/webapps.md
Files
master
Go to file
.github
12factor
Docker-Orchestration
beginner
chapters
alpine.md
setup.md
votingapp.md
webapps.md
flask-app
images
static-site
readme.md
developer-tools
dockercon-us-2017
networking
raspberrypi
registry
security
slides
swarm-mode
windows
.gitignore
.gitmodules
labs / beginner / chapters / webapps.md
Top
Preview
Code
Blame
498 lines (358 loc) · 24.7 KB
Raw
Actually, you probably won't be able to answer any of these questions yet! In this case, the client didn't tell the Docker Engine to publish any of the ports, so you need to re-run the docker run command to add this instruction.
Let's re-run the command with some new flags to publish ports and pass your name to the container to customize the message displayed. We'll use the -d option again to run the container in detached mode.
First, stop the container that you have just launched. In order to do this, we need the container ID.
Since we ran the container in detached mode, we don't have to launch another terminal to do this. Run docker ps to view the running containers.
$ docker ps
CONTAINER ID IMAGE COMMAND
a7a0e504ca3e dockersamples/static-site "/bin/sh -c 'cd /u
Check out the CONTAINER ID column. You will need to use this CONTAINER I
container you want to stop, and then to remove it. The example below provide
that you see in your terminal.
$ docker stop a7a0e504ca3e
$ docker rm a7a0e504ca3e
Note: A cool feature is that you do not need to specify the entire CONTAIN
is unique among all the containers that you have launched, the Docker clie
Now, let's launch a container in detached mode as shown below:
$ docker run --name static-site -e AUTHOR="Your Name" -d -P dockersamples/static-site
e61d12292d69556eabe2a44c16cbd54486b2527e2ce4f95438e504afb7b02810
In the above command:
-d will create a container with the process detached from our terminal
-P will publish all the exposed container ports to random ports on the Docker host
wahajhaider — -zsh — 80×24
STATUS PORTS NAMES
a7a0e504ca3e dockersamples/static-site "/bin/sh -c
dquote>
wahajhaider@Wahajs-MacBook-Pro ~ % docker ps
CONTAINER ID IMAGE COMMAND CREATED
STATUS PORTS NAMES
4cdea7a386d3 dockersamples/static-site "/bin/sh -c 'cd /usr…" 9 minutes ag
o Up 9 minutes 80/tcp, 443/tcp stupefied_hei
senberg
206028173d7b dockersamples/static-site "/bin/sh -c 'cd /usr…" 27 hours ago
Up 27 hours 0.0.0.0:55001->80/tcp, 0.0.0.0:55000->443/tcp static-site
bc1691501759 dockersamples/static-site "/bin/sh -c 'cd /usr…" 27 hours ago
Up 27 hours 80/tcp, 443/tcp jovial_wozni
k
wahajhaider@Wahajs-MacBook-Pro ~ % docker stop 4cd
4cd
wahajhaider@Wahajs-MacBook-Pro ~ % docker ps
CONTAINER ID IMAGE COMMAND CREATED
STATUS PORTS NAMES
206028173d7b dockersamples/static-site "/bin/sh -c 'cd /usr…" 27 hours ago
Up 27 hours 0.0.0.0:55001->80/tcp, 0.0.0.0:55000->443/tcp static-site
bc1691501759 dockersamples/static-site "/bin/sh -c 'cd /usr…" 27 hours ago
Up 27 hours 80/tcp, 443/tcp jovial_wozniak
wahajhaider@Wahajs-MacBook-Pro ~ %

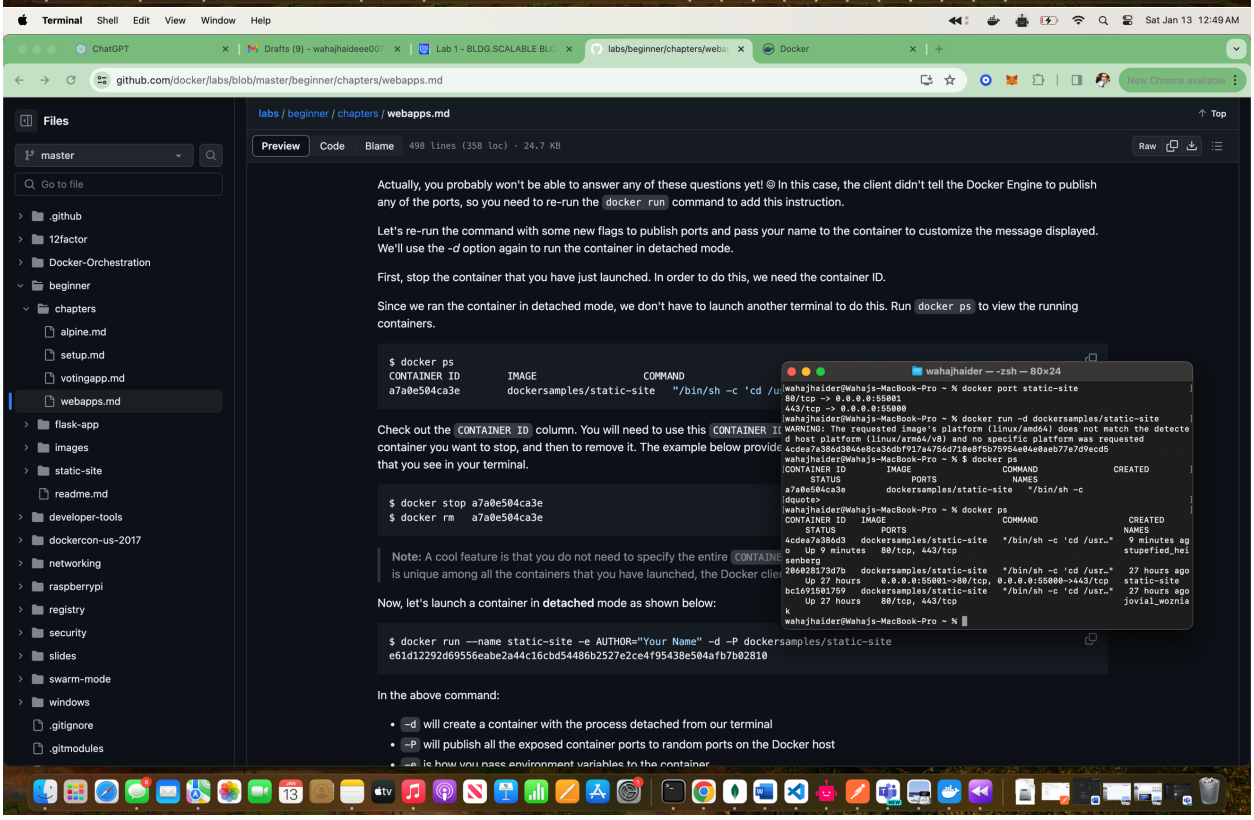

Terminal Shell Edit View Window Help
Thu Jan 11 8:50PM
Lab 1 - BLDG.SCALABLE BLC
labs/beginner/chapters/alpine
github.com/docker/labs/blob/master/beginner/chapters/alpine.md
wahajhaider — -zsh — 80×24
Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/
For more examples and ideas, visit:
https://docs.docker.com/get-started/
wahajhaider@Wahajs-MacBook-Pro ~ % $ docker pull alpine
zsh: command not found: $
wahajhaider@Wahajs-MacBook-Pro ~ % docker pull alpine
Using default tag: latest
latest: Pulling from library/alpine
c30352492317: Pull complete
Digest: sha256:51b67269f354137895d43f3b3d810bfacd3945438e94dc5ac55fdac340352f48
Status: Downloaded newer image for alpine:latest
docker.io/library/alpine:latest
What's Next?
View a summary of image vulnerabilities and recommendations → docker scout quickview alpine
wahajhaider@Wahajs-MacBook-Pro ~ % docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
alpine latest 1dc785547989 5 weeks ago 7.73MB
hello-world latest ee301c921b8a 8 months ago 9.14kB
wahajhaider@Wahajs-MacBook-Pro ~ %
Files
master
Go to file
.github
12factor
Docker-Orchestration
beginner
chapters
alpine.md
setup.md
votingapp.md
webapps.md
flask-app
images
static-site
readme.md
developer-tools
dockercon-us-2017
networking
raspberrypi
registry
security
slides
swarm-mode
windows
.gitignore
.gitmodules
labs / beginner / chapters / alpine.md
Preview Code Blame 99 lines (78 loc) · 7.17 KB
Top
$ docker pull alpine
Note: Depending on how you've installed docker on your system, you
command. Try the commands from the Getting Started tutorial to ver
docker commands with sudo. Alternatively you can create a docke
The pull command fetches the alpine image from the Docker registry
command to see a list of all images on your system.
$ docker images
REPOSITORY TAG IMAGE ID CREATED VIRTUAL SIZE
alpine latest c51f86c28340 4 weeks ago 1.109 MB
hello-world latest 690ed74de00f 5 months ago 960 B
1.1 Docker Run
Great! Let's now run a Docker container based on this image. To do that you are going to use the docker run command.
$ docker run alpine ls -l
total 48
drwxr-xr-x 2 root root 4096 Mar 2 16:20 bin
drwxr-xr-x 5 root root 360 Mar 18 09:47 dev
drwxr-xr-x 13 root root 4096 Mar 18 09:47 etc
drwxr-xr-x 2 root root 4096 Mar 2 16:20 home
drwxr-xr-x 5 root root 4096 Mar 2 16:20 lib
.....
.....
What happened? Behind the scenes, a lot of stuff happened. When you call run,
1. The Docker client contacts the Docker daemon
2. The Docker daemon checks local store if the image (alpine in this case) is available locally, and if not, downloads it from Docker Store. (Since we have issued docker pull alpine before, the download step is not necessary)
3. The Docker daemon creates the container and then runs a command in that container.
4. The Docker daemon streams the output of the command to the Docker client

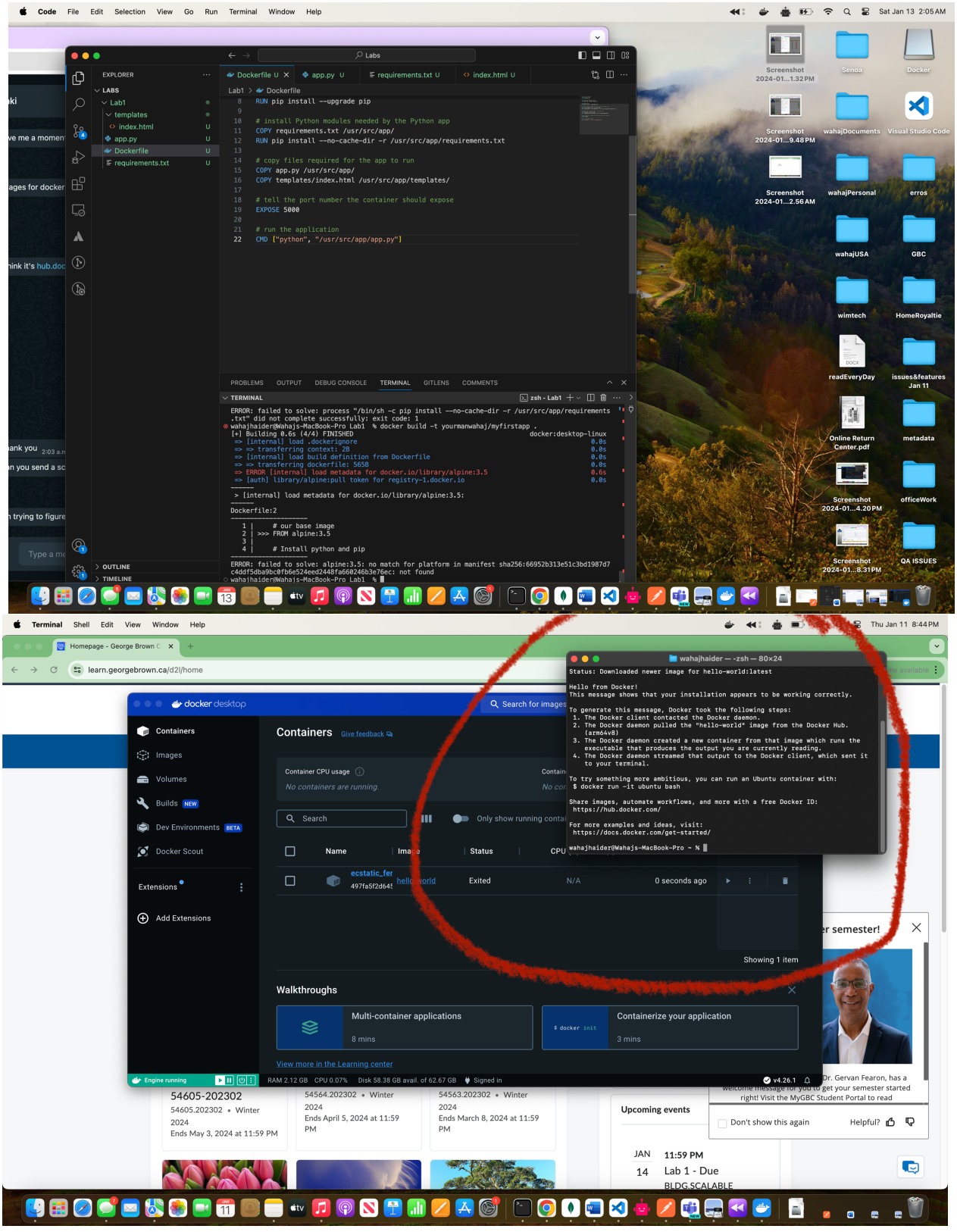

Terminal Shell Edit View Window Help
Thu Jan 11 8:57PM
Lab 1 - BLDG.SCALABLE BLC
labs/beginner/chapters/alpine
github.com/docker/labs/blob/master/beginner/chapters/alpine.md
wahajhaider — -zsh — 80×24
drwxr-xr-x 2 root root 4096 Dec 7 09:45 bin
drwxr-xr-x 5 root root 340 Jan 12 01:53 dev
drwxr-xr-x 1 root root 4096 Jan 12 01:53 etc
drwxr-xr-x 2 root root 4096 Dec 7 09:45 home
drwxr-xr-x 7 root root 4096 Dec 7 09:45 lib
drwxr-xr-x 5 root root 4096 Dec 7 09:45 media
drwxr-xr-x 2 root root 4096 Dec 7 09:45 mnt
drwxr-xr-x 2 root root 4096 Dec 7 09:45 opt
dr-xr-xr-x 265 root root 0 Jan 12 01:53 proc
drwx------ 2 root root 4096 Dec 7 09:45 root
drwxr-xr-x 2 root root 4096 Dec 7 09:45 run
drwxr-xr-x 2 root root 4096 Dec 7 09:45 sbin
drwxr-xr-x 2 root root 4096 Dec 7 09:45 srv
dr-xr-xr-x 12 root root 0 Jan 12 01:53 sys
drwxrwxrwt 2 root root 4096 Dec 7 09:45 tmp
drwxr-xr-x 7 root root 4096 Dec 7 09:45 usr
drwxr-xr-x 12 root root 4096 Dec 7 09:45 var
wahajhaider@Wahajs-MacBook-Pro ~ % $ docker run alpine echo "hello from alpine"
zsh: command not found: $
wahajhaider@Wahajs-MacBook-Pro ~ % docker run alpine echo "hello from alpine"
hello from alpine
wahajhaider@Wahajs-MacBook-Pro ~ %
Files
master
Go to file
.github
12factor
Docker-Orchestration
beginner
chapters
alpine.md
setup.md
votingapp.md
webapps.md
flask-app
images
static-site
readme.md
developer-tools
dockercon-us-2017
networking
raspberrypi
registry
security
slides
swarm-mode
windows
.gitignore
.gitmodules
labs / beginner / chapters / alpine.md
Preview Code Blame 99 lines (78 loc) · 7.17 KB
Top
1. The Docker client contacts the Docker daemon
2. The Docker daemon checks local store if the image (alpine in this
(Since we have issued docker pull alpine before, the download
3. The Docker daemon creates the container and then runs a comma
4. The Docker daemon streams the output of the command to the Do
When you run docker run alpine, you provided a command (ls -l)
Let's try something more exciting.
$ docker run alpine echo "hello from alpine"
hello from alpine
OK, that's some actual output. In this case, the Docker client dutifully ran the echo command in our alpine container and then exited it. If you've noticed, all of that happened pretty quickly. Imagine booting up a virtual machine, running a command and then killing it. Now you know why they say containers are fast!
Try another command.
$ docker run alpine /bin/sh
Wait, nothing happened! Is that a bug? Well, no. These interactive shells will exit after running any scripted commands, unless they are run in an interactive terminal - so for this example to not exit, you need to docker run -it alpine /bin/sh.
You are now inside the container shell and you can try out a few commands like ls -l, uname -a and others. Exit out of the container by giving the exit command.
Ok, now it's time to see the docker ps command. The docker ps command shows you all containers that are currently running.
$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
Since no containers are running, you see a blank line. Let's try a more useful variant: docker ps -a
$ docker ps -a


Terminal Shell Edit View Window Help
Thu Jan 11 9:02PM
github.com/docker/labs/blob/master/beginner/chapters/alpine.md
Files
labs / beginner / chapters / alpine.md
Preview Code Blame 99 lines (78 loc) · 7.17 KB
When you run docker run alpine, you provided a command (ls -l)
Let's try something more exciting.
$ docker run alpine echo "hello from alpine"
hello from alpine
OK, that's some actual output. In this case, the Docker client dutifully r
you've noticed, all of that happened pretty quickly. Imagine booting up
know why they say containers are fast!
Try another command.
$ docker run alpine /bin/sh
Wait, nothing happened! Is that a bug? Well, no. These interactive shells will exit after running any scripted commands, unless they are run in an interactive terminal - so for this example to not exit, you need to docker run -it alpine /bin/sh.
You are now inside the container shell and you can try out a few commands like ls -l, uname -a and others. Exit out of the container by giving the exit command.
Ok, now it's time to see the docker ps command. The docker ps command shows you all containers that are currently running.
$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
Since no containers are running, you see a blank line. Let's try a more useful variant: docker ps -a
$ docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORT
36171a5da744 alpine "/bin/sh" 5 minutes ago Exited (0) 2 minutes ago
a6a9d46d0b2f alpine "echo 'hello from alp" 6 minutes ago Exited (0) 6 minutes ago
ff0a5c3750b9 alpine "ls -l" 8 minutes ago Exited (0) 8 minutes ago
c317d0a9e3d2 hello-world "/hello" 34 seconds ago Exited (0) 12 minutes ago
wahajhaider — -zsh — 80×24
/ # ls
/ # ^C
/ # exit
wahajhaider@Wahajs-MacBook-Pro ~ % docker ps
wahajhaider@Wahajs-MacBook-Pro ~ %


Terminal Shell Edit View Window Help
Thu Jan 11 9:05PM
github.com/docker/labs/blob/master/beginner/chapters/alpine.md
Files
labs / beginner / chapters / alpine.md
Preview Code Blame 99 lines (78 loc) · 7.17 KB
Since no containers are running, you see a blank line. Let's try a more u
$ docker ps -a
CONTAINER ID IMAGE COMMAND
36171a5da744 alpine "/bin/sh"
a6a9d46d0b2f alpine "echo 'hello from alp'
ff0a5c3750b9 alpine "ls -l"
c317d0a9e3d2 hello-world "/hello"
What you see above is a list of all containers that you ran. Notice that the STATUS column shows that these containers exited a few minutes ago. You're probably wondering if there is a way to run more than just one command in a container. Let's try that now:
$ docker run -it alpine /bin/sh
/ # ls
bin dev etc home lib linuxrc media mnt proc root run sbin sys
/ # uname -a
Linux 97916e8cb5dc 4.4.27-moby #1 SMP Wed Oct 26 14:01:48 UTC 2016 x86_64 Linux
Running the run command with the -it flags attaches us to an interactive tty in the container. Now you can run as many commands in the container as you want. Take some time to run your favorite commands.
That concludes a whirlwind tour of the docker run command which would most likely be the command you'll use most often. It makes sense to spend some time getting comfortable with it. To find out more about run, use docker run --help to see a list of all flags it supports. As you proceed further, we'll see a few more variants of docker run.
1.2 Terminology
In the last section, you saw a lot of Docker-specific jargon which might be confusing to some. So before you go further, let's clarify some terminology that is used frequently in the Docker ecosystem.
Images - The file system and configuration of our application which are used to create containers. To find out more about a Docker image, run docker inspect alpine. In the demo above, you used the docker pull command to download the alpine image. When you executed the command docker run hello-world, it also did a docker pull behind the scenes to download the hello-world image.
wahajhaider — -zsh — 80×24
/ # ls
/ # ^C
/ # exit
wahajhaider@Wahajs-MacBook-Pro ~ % docker ps
wahajhaider@Wahajs-MacBook-Pro ~ % docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
561478b19914 alpine "/bin/sh" 3 minutes ago Exited (1
30) 3 minutes ago charming_beaver
c3fbef643fd1 alpine "/bin/sh" 4 minutes ago Exited (0
) 4 minutes ago magical_kilby
0e015d699834 alpine "/bin/sh" 5 minutes ago Exited (0
) 5 minutes ago determined_colden
5c5989396869 alpine "echo 'hello from al…" 9 minutes ago Exited (0
) 9 minutes ago adoring_engelbart
312d6c49bca1 alpine "ls -l" 11 minutes ago Exited (0
) 11 minutes ago priceless_newton
497fa5f2d665 hello-world "/hello" 21 minutes ago Exited (0
) 21 minutes ago ecstatic_fermi
wahajhaider@Wahajs-MacBook-Pro ~ %


Terminal Shell Edit View Window Help
Thu Jan 11 9:10PM
Lab 1 - BLDG.SCALABLE BLC
labs/beginner/chapters/alpine
github.com/docker/labs/blob/master/beginner/chapters/alpine.md
wahajhaider — com.docker.cli • docker run -it alpine /bin/sh — 80×24
/ # exit
wahajhaider@Wahajs-MacBook-Pro ~ % docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
wahajhaider@Wahajs-MacBook-Pro ~ % docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
561478b19914 alpine "/bin/sh" 3 minutes ago Exited (130) 3 minutes ago charming_beaver
c3fbef643fd1 alpine "/bin/sh" 4 minutes ago Exited (0) 4 minutes ago magical_kilby
0e815d699834 alpine "/bin/sh" 5 minutes ago Exited (0) 5 minutes ago determined_colden
5c5989396869 alpine "echo 'hello from al…" 9 minutes ago Exited (0) 9 minutes ago adoring_engelbart
312d6c49bca1 alpine "ls -l" 11 minutes ago Exited (0) 11 minutes ago priceless_newton
497fa5f2d645 hello-world "/hello" 21 minutes ago Exited (0) 21 minutes ago ecstatic_fermi
wahajhaider@Wahajs-MacBook-Pro ~ % docker run -it alpine /bin/sh
/ # uname -a
Linux f732ffc28b17 6.5.11-linuxkit #1 SMP PREEMPT Wed Dec 6 17:08:31 UTC 2023 aarch64 Linux
/ #
Files
master
Go to file
.github
12factor
Docker-Orchestration
beginner
chapters
alpine.md
setup.md
votingapp.md
webapps.md
flask-app
images
static-site
readme.md
developer-tools
dockercon-us-2017
networking
raspberrypi
registry
security
slides
swarm-mode
windows
.gitignore
.gitmodules
labs / beginner / chapters / alpine.md
Top
Preview Code Blame 99 lines (78 loc) · 7.17 KB
What you see above is a list of all containers that you ran. Notice that t
ago. You're probably wondering if there is a way to run more than just c
$ docker run -it alpine /bin/sh
/ # ls
bin dev etc home lib linuxrc media
/ # uname -a
Linux 97916e8cb5dc 4.4.27-moby #1 SMP Wed Oct 26 14:01:48 UTC
Running the run command with the -it flags attaches us to an interactive tty in the container. Now you can run as many commands in the container as you want. Take some time to run your favorite commands.
That concludes a whirlwind tour of the docker run command which would most likely be the command you'll use most often. It makes sense to spend some time getting comfortable with it. To find out more about run, use docker run --help to see a list of all flags it supports. As you proceed further, we'll see a few more variants of docker run.
1.2 Terminology
In the last section, you saw a lot of Docker-specific jargon which might be confusing to some. So before you go further, let's clarify some terminology that is used frequently in the Docker ecosystem.
Images - The file system and configuration of our application which are used to create containers. To find out more about a Docker image, run docker inspect alpine. In the demo above, you used the docker pull command to download the alpine image. When you executed the command docker run hello-world, it also did a docker pull behind the scenes to download the hello-world image.
Containers - Running instances of Docker images — containers run the actual applications. A container includes an application and all of its dependencies. It shares the kernel with other containers, and runs as an isolated process in user space on the host OS. You created a container using docker run which you did using the alpine image that you downloaded. A list of running containers can be seen using the docker ps command.
Docker daemon - The background service running on the host that manages building, running and distributing Docker containers.
Docker client - The command line tool that allows the user to interact with the Docker daemon.
Docker Store - A registry of Docker images, where you can find trusted and enterprise ready containers, plugins, and Docker editions. You'll be using this later in this tutorial.
Next Steps


Terminal Shell Edit View Window Help
Sat Jan 13 12:49 AM
ChatGPT
Drafts (9) - wahajhaideee007
Lab 1 - BLDG.SCALABLE BLC
labs/beginner/chapters/webapps
Docker
github.com/docker/labs/blob/master/beginner/chapters/webapps.md
wahajhaider — -zsh — 80×24
wahajhaider@Wahajs-MacBook-Pro ~ % docker port static-site
80/tcp -> 0.0.0.0:55001
443/tcp -> 0.0.0.0:55000
wahajhaider@Wahajs-MacBook-Pro ~ % docker run -d dockersamples/static-site
WARNING: The requested image's platform (linux/amd64) does not match the detected host platform (linux/arm64/v8) and no specific platform was requested
4cdea7a386d3044e8ca36dbf917a4754d710e8f5b75954e04e0aeb77e7e9ecd5
wahajhaider@Wahajs-MacBook-Pro ~ % $ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
a7a0e504ca3e dockersamples/static-site "/bin/sh -c 'cd /usr…" 9 minutes ago Up 9 minutes 80/tcp, 443/tcp stupefied_heisenberg
206028173d7b dockersamples/static-site "/bin/sh -c 'cd /usr…" 27 hours ago Up 27 hours 0.0.0.0:55001->80/tcp, 0.0.0.0:55000->443/tcp static-site
bc1691501759 dockersamples/static-site "/bin/sh -c 'cd /usr…" 27 hours ago Up 27 hours 80/tcp, 443/tcp jovial_wozniak
wahajhaider@Wahajs-MacBook-Pro ~ %
Files
master
Go to file
.github
12factor
Docker-Orchestration
beginner
chapters
alpine.md
setup.md
votingapp.md
webapps.md
flask-app
images
static-site
readme.md
developer-tools
dockercon-us-2017
networking
raspberrypi
registry
security
slides
swarm-mode
windows
.gitignore
.gitmodules
labs / beginner / chapters / webapps.md
Top
Preview Code Blame 498 lines (358 loc) · 24.7 KB
Raw
Actually, you probably won't be able to answer any of these questions yet! ☺ In this case, the client didn't tell the Docker Engine to publish any of the ports, so you need to re-run the docker run command to add this instruction.
Let's re-run the command with some new flags to publish ports and pass your name to the container to customize the message displayed. We'll use the -d option again to run the container in detached mode.
First, stop the container that you have just launched. In order to do this, we need the container ID.
Since we ran the container in detached mode, we don't have to launch another terminal to do this. Run docker ps to view the running containers.
$ docker ps
CONTAINER ID IMAGE COMMAND
a7a0e504ca3e dockersamples/static-site "/bin/sh -c 'cd /u
Check out the CONTAINER ID column. You will need to use this CONTAINER I
container you want to stop, and then to remove it. The example below provide
that you see in your terminal.
$ docker stop a7a0e504ca3e
$ docker rm a7a0e504ca3e
Note: A cool feature is that you do not need to specify the entire CONTAIN
is unique among all the containers that you have launched, the Docker clie
Now, let's launch a container in detached mode as shown below:
$ docker run --name static-site -e AUTHOR="Your Name" -d -P dockersamples/static-site
e61d12292d69556eabe2a44c16cbd54486b2527e2ce4f95438e504afb7b02810
In the above command:
-d will create a container with the process detached from our terminal
-P will publish all the exposed container ports to random ports on the Docker host
-e is how you pass environment variables to the container


Code File Edit Selection View Go Run Terminal Window Help
Sat Jan 13 2:05 AM
EXPLORER
LABS
Lab1
templates
index.html
app.py
Dockerfile
requirements.txt
Dockerfile app.py requirements.txt index.html
RUN pip install --upgrade pip
install Python modules needed by the Python app
COPY requirements.txt /usr/src/app/
RUN pip install --no-cache-dir -r /usr/src/app/requirements.txt
copy files required for the app to run
COPY app.py /usr/src/app/
COPY templates/index.html /usr/src/app/templates/
tell the port number the container should expose
EXPOSE 5000
run the application
CMD ["python", "/usr/src/app/app.py"]
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL GITLENS COMMENTS
ERROR: failed to solve: process "/bin/sh -c pip install --no-cache-dir -r /usr/src/app/requirements.txt" did not complete successfully: exit code: 1
wahajhaider@Wahajs-MacBook-Pro Lab1 % docker build -t yourmanwahaj/myfirstapp .
[+] Building 0.6s (4/4) FINISHED
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 565B
=> ERROR [internal] load metadata for docker.io/library/alpine:3.5
=> [auth] library/alpine:pull token for registry-1.docker.io
> [internal] load metadata for docker.io/library/alpine:3.5:
Dockerfile:2
1 | # our base image
2 | >>> FROM alpine:3.5
3 |
4 | # Install python and pip
ERROR: failed to solve: alpine:3.5: no match for platform in manifest sha256:66952b313e51c3bd1987d7c4ddf5dba9bc0fb6e524eed2448fa660246b3e76ec: not found
wahajhaider@Wahajs-MacBook-Pro Lab1 %
Screenshot 2024-01...1.32 PM
Senoa
Docker
Screenshot 2024-01...9.48 PM
wahajDocuments
Visual Studio Code
Screenshot 2024-01...2.56 AM
wahajPersonal
erros
wahajUSA
GBC
wimtech
HomeRoyaltie
readEveryDay
issues&features Jan 11
Online Return Center.pdf
metadata
Screenshot 2024-01...4.20 PM
officeWork
Screenshot 2024-01...8.31 PM
QA ISSUES
Terminal Shell Edit View Window Help
Thu Jan 11 8:44 PM
Homepage - George Brown C
learn.georgebrown.ca/d2l/home
wahajhaider — -zsh — 80×24
Status: Downloaded newer image for hello-world:latest
Hello from Docker!
This message shows that your installation appears to be working correctly.
To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub. (arm64v8)
3. The Docker daemon created a new container from that image which runs the executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it to your terminal.
To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash
Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/
For more examples and ideas, visit:
https://docs.docker.com/get-started/
wahajhaider@Wahajs-MacBook-Pro ~ %
docker desktop
Search for images
Containers
Images
Volumes
Builds NEW
Dev Environments BETA
Docker Scout
Extensions
Add Extensions
Containers Give feedback
Container CPU usage
No containers are running.
Search
Only show running containers
Name Image Status CPU
ecstatic_fer 497fa5f2d64 hello-world Exited N/A 0 seconds ago
Showing 1 item
Walkthroughs
Multi-container applications 8 mins
Containerize your application 3 mins
View more in the Learning center
Engine running
RAM 2.12 GB CPU 0.07% Disk 58.38 GB avail. of 62.67 GB Signed in
v4.26.1
semester!
Dr. Gervan Fearon, has a welcome message for you to get your semester started right! Visit the MyGBC Student Portal to read
Don't show this again
Helpful?
54605-202302
54605.202302 • Winter 2024
Ends May 3, 2024 at 11:59 PM
54564.202302 • Winter 2024
Ends April 5, 2024 at 11:59 PM
54563.202302 • Winter 2024
Ends March 8, 2024 at 11:59 PM
Upcoming events
JAN 14
11:59 PM
Lab 1 - Due
BLDG.SCALABLE