Docker and friends Cheat Sheet

by Andrew Matthews (aabs) via cheatography.com/22451/cs/4601/

Building Images

`docker build`	Build a new image from the source code at PATH `docker build -t myimg`
`-q`	Suppress the output generated by containers
`--rm`	Remove intermediate containers after a successful build
`-t`	Repository name (and optionally a tag) for the image

Examples:

Simplest possible build instruction:

`docker build .`

Name image and tag as v1.5:

`docker build -t myorg/ myi mg:1. 5 .`

Misc useful commands

Misc useful commands (cont)

top	Lookup the running processes of a container
wait	Block until a container stops, then print its exit code

Run `docker COMMAND --help` for help on the command.

Running Docker

`docker run`	starts a process with its own file system, its own networking, and its own isolated process tree `docker run -itP image`
`--name`	name the container `docker run --name =so mename o rg`
`-t`	terminal interface
`-i`	interactive session
`-d`	daemon Mode
`-P`	publish all exposed ports
`-p`	expose specific port `-p ip:hos tpo rt: con tai nerp ort`
`--rm`	remove intermediate images
`-v`	bind mount a volume `-v /host: /co ntainer`

Examples:

run an interactive session:

`sudo docker run -P --name =so me name image`

run container in background, on port 80:

`sudo docker run --d -v /etc/a pp d ata :/data -p 0.0.0.0 :8 0:808 0 --rm nginx`

Docker Compose

TBC

Dockerfiles (for creating images)

Dockerfiles (for creating images) (cont)

`EXPOSE`	listen on the network ports `EXPOSE <po rt> [<p ort`

`ENV`	set the environment variables
	`ENV <ke y> <va lue>`
	`ENV <ke y>= <va lue> <`
	`a lue>`

`ADD` / `COPY` ??	**copies** new files from `<sr c>` them to the container at `<de` `ADD <sr c>... <de st>`
`ENTRYPOINT`	configure container to run as a executable `ENTRYPOINT ["ex e", "`
`VOLUME`	create externally mounted vol `VOLUME ["/d ata "]`
`USER`	sets the user name or UID to running the image `USER<u ser>`
`WORKDIR`	sets working dir for any `RUN`, `(` `POINT`, `COPY` and `ADD` instruc `WORKDIR /path/ to/ wor`
`ONBUILD`	adds a trigger instruction for e when image is used as base c build `ONBUILD [INSTR UCTION]`

Check out the Manual Page for more detail.

attach	Attach to a running container
cp	Copy files/folders from a container's filesystem to the host path
create	Create a new container
exec	Run a command in a running container
images	List images
export	Stream the contents of a container as a tar archive
import	Create a new filesystem image from the contents of a tarball
inspect	Return low-level information on a container or image
kill	Kill a running container
load	Load an image from a tar archive
logs	Fetch the logs of a container
port	Lookup the public-facing port that is NAT-ed to PRIVATE_PORT
ps	List containers
pull	Pull an image or a repository from a Docker registry server
push	Push an image or a repository to a Docker registry server
rm	Remove one or more containers
rmi	Remove one or more images
save	Save an image to a tar archive
search	Search for an image on the Docker Hub
start	Start a stopped container
stop	Stop a running container

FROM	set base image for this image FROM <im age >:< tag>
MAINTAINER	The author of the image MAINTAINER <na me>
RUN	execute commands in new layer RUN <co mma nd> RUN ["ex ecu tab le", " par am1 " , ...]
CMD	provide default for an executing container CMD ["ex ecu tab le", " par am1 " ,...]
LABEL	add metadata to image LABEL <ke y>= <va lue> <ke y>= <v a lue>

By **Andrew Matthews** (aabs)
cheatography.com/aabs/
aabs.wordpress.com

Published 15th July, 2015.
Last updated 12th May, 2016.