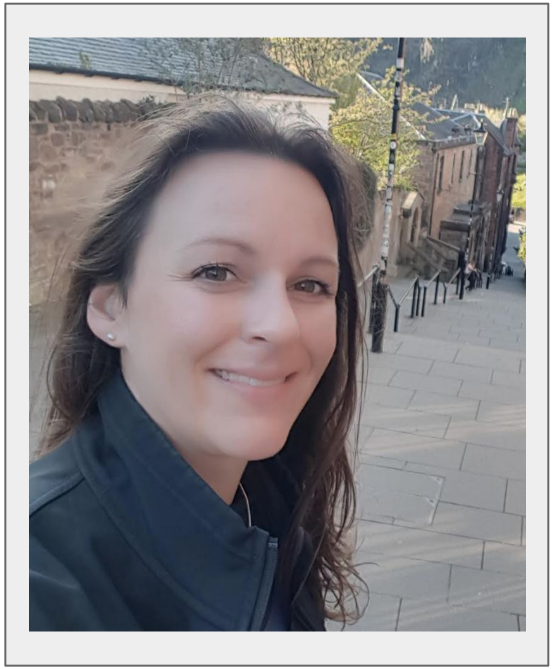
Dockerfile Dos & Do nOts

There are many
ways to
skin a cat…

but *SHOULD* you???

Melissa McKay
Developer Advocate
JFrog

About Melissa McKay...



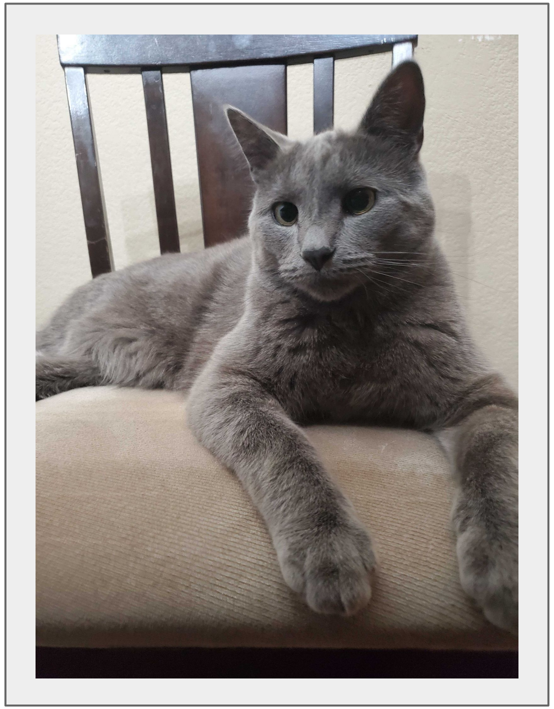
Developer Advocate @JFrog

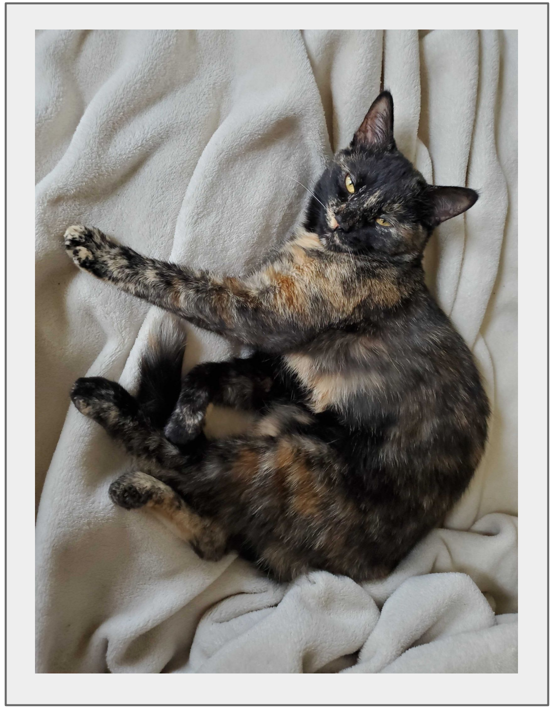
Java Champion
Docker Captain

TWITTER:
@melissajmckay

LINKEDIN:
linkedin.com/in/melissajmckay

Buzz





Bee

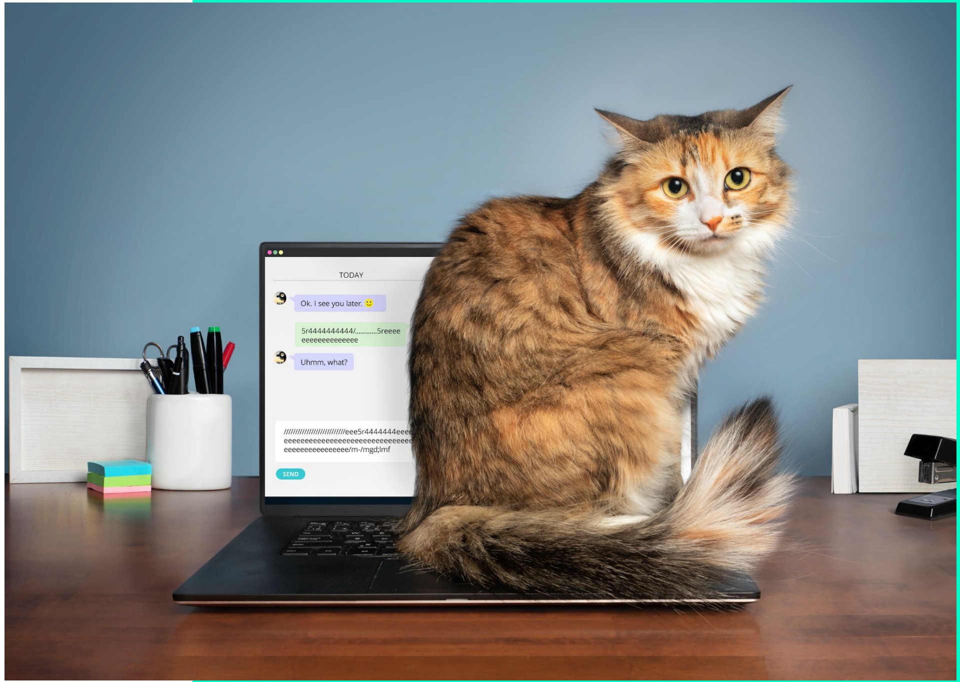
The Dockerfile

FROM	ENV	STOPSIGNAL
ADD	ARG	ONBUILD
COPY	WORKDIR	SHELL
RUN	LABEL	HEALTHCHECK
USER	EXPOSE	ENTRYPOINT
CMD	VOLUME	

https://docs.docker.com/engine/reference/builder/

8 DO NOTS

1) IGNORING .DOCKERIGNORE


TODAY

Ok. I see you later. 🙂

5r4444444444/............5reeee eeeeeeeeeeeeee

Uhmm, what?

//////////////////////////eee5r4444444eeee eeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee eeeeeeeeeeeeeeeee/m-/mgd;lmf

```
FROM ubuntu

WORKDIR /myapp

COPY . /myapp

EXPOSE 8080

ENTRYPOINT [“start.sh”]
```

WHY USE .DOCKERIGNORE?

Avoid wasted time and invalidating cache by sending EVERYTHING to the Docker daemon

Avoid sending test or user specific files

Avoid sending secrets!

DO

```
# Ignore these files in my project

**/*.md

!README.md

passwords.txt

.git

logs/

*/temp

**/test/
```

File: .dockerignore

DO NOT

```
404
```

File: .dockerignore

2) Using untrusted base images

```
FROM evilimage

WORKDIR /myapp

COPY . /myapp

ENTRYPOINT [“start.sh”]
```

WHY USE TRUSTED BASE IMAGES?

Evaluate the image for your use case – KNOW WHAT’S IN IT!

Avoid malicious packages

Get latest updates

Start with Docker Official Images – mitigate your risk

3) NEVER UPDATING

```
FROM baseimage:2-years-ago

WORKDIR /myapp

COPY . /myapp

ENTRYPOINT [“start.sh”]
```

WHY UPDATE?

Security updates are important!

Security updates are really important!

Security updates are really, REALLY important!

4)
NOT DEFINING VERSIONS

```
FROM mybaseimage

RUN apt-get update \
   && install -y \
    mypackage
    anotherpackage
    yetanotherpackage

WORKDIR /myapp

COPY . /myapp

EXPOSE 8080

ENTRYPOINT ["/start.sh"]
```

WHY DEFINE VERSIONS?

Have a bill of materials for your build - know what version of EVERYTHING is installed

Save yourself troubleshooting time by explicitly controlling version updates

5) INCLUDING BUILD TOOLS

```
FROM maven:3.6.3-jdk-8

WORKDIR /myapp

COPY . /myapp

RUN mvn clean package

ENTRYPOINT [“start.sh”]
```

WHY NOT INCLUDE YOUR BUILD TOOL?

The size of your image will be bigger than it needs to be

Minimize your attack surface area by ONLY including what you need for your application to run in production

You can use a multi-stage build instead!

6) Using External Resources

```
FROM ubuntu

RUN apt-get update \
    && install -y curl

RUN curl -sL \
https://somewhere.com/script.sh \
  | bash -

WORKDIR /myapp

COPY . /myapp

ENTRYPOINT [“start.sh”]
```

WHY NOT USE EXTERNAL RESOURCES?

If an external resource goes away… what do you do???

Not reviewing external scripts before they are used in your production environment is an excellent opening for a supply chain attack.

7) Hardcoding Secrets or config

```
FROM mybaseimage

RUN apt-get update

RUN rm -rf secrets

WORKDIR /myapp

COPY . /myapp

EXPOSE 8080
ENTRYPOINT [“/start.sh”]
```

WHY AVOID HARDCODED SECRETS OR CONFIG?

It’s never a good idea to advertise sensitive information in artifacts that will be moved around, possibly replicated, and deployed into production (or anywhere else)

Providing configuration at runtime allows for images to be environment agnostic

8) Doing Too Much!

```
FROM mybaseimage:1.0.0

RUN sudo apt-get purge \
    --auto-remove oldpackage

RUN apt-get update \
    && apt-get install -y \
    newpackage

WORKDIR /myapp

COPY . /myapp

RUN /cleanupdatabase.sh

RUN /run_unit_tests.sh

ENTRYPOINT [“start.sh”]
```

WHY KEEP IT SIMPLE?

Dockerfile should describe the build

If the base image needs modified - modify it!

Dockerfiles should contain idempotent operations only - in order to provide repeatable builds

RESOURCES

DOCKERFILE DOCUMENTATION
https://docs.docker.com/engine/reference/builder/

OFFICIAL IMAGES
https://docs.docker.com/docker-hub/official_images/

MULTI-STAGE BUILDS
https://docs.docker.com/develop/develop-images/multistage-build/

STORING YOUR IMAGES
https://dzone.com/refcardz/getting-started-with-container-registries

THANK YOU!
Q & A