

# LEVEL UP

## Your Java Container Images

Melissa McKay  
Developer Advocate @JFrog



# THE SWAG

**WIN**  
A JFROG SHIRT!



<http://bit.ly/JFokusJFrog2021>





# MELISSA MCKAY

Developer Advocate @JFrog



@melissajmckay



[linkedin.com/in/melissajmckay](https://www.linkedin.com/in/melissajmckay)

# THE AGENDA

- Brief History
- The Container Market
- What is Docker?
- What is a Container?
- Container Gotchas



# HOW ARE YOU USING CONTAINERS TODAY???

- LOCALLY
- TEST/QA ENVIRONMENTS
- PRODUCTION
- WE DON'T USE THEM TODAY
- WE ARE CONSIDERING USING THEM



**ALL  
ABOUT...**

**CONTAINERS**





# SHARING LIMITED RESOURCES



1979 / 1982- chroot

# PROGRESS TOWARD VIRTUALIZATION

- 2000 - FreeBSD jail
- 2004 - Solaris Zones / snapshots
- 2006 - Google Process Containers / cgroups
- 2008 - IBM LinuX Containers (LXC)
- 2013 - Docker (open source!)
  - Google LMCTFY (open source!)
- 2014 - Docker trades LXC for libcontainer
- ... *more stuff happened*
- June 2015 - Open Container Project/Initiative (OCI)
  - Runtime Specification (runtime-spec)
  - Image Specification (image-spec)
- ... *even more stuff happened and is **still happening!***

2011  
Java 7

2014  
Java 8





# THE CONTAINER MARKET (according to Sysdig)

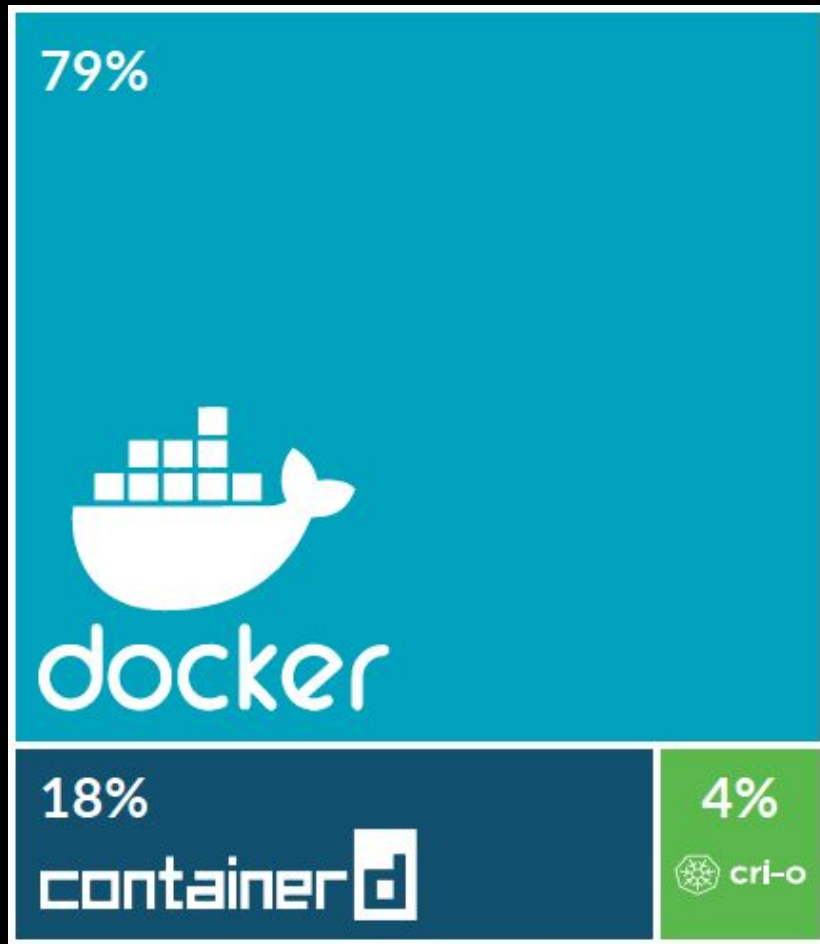
2017 - 45,000 Containers, 99% Docker

2018 - 90,000 Containers



Fig. 1. 2018 Container Runtimes from: "2018 Docker usage report," 29 May. 2018, [sysdig.com/blog/2018-docker-usage-report/](https://sysdig.com/blog/2018-docker-usage-report/). Accessed 10 Jun. 2020.

# THE CONTAINER MARKET (according to Sysdig)

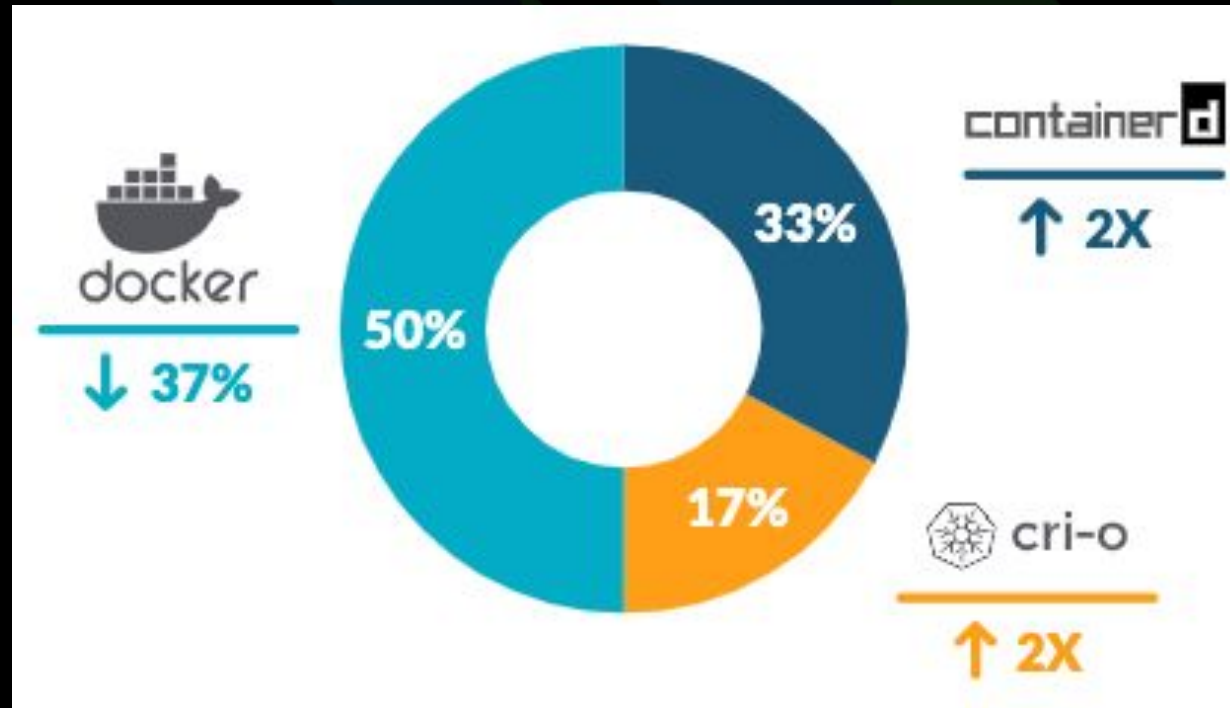


2019 - 2 million Containers  
(includes both SaaS & on prem users)

**Fig. 2.** 2019 Container Runtimes from: "Sysdig 2019 Container Usage Report: New Kubernetes and security insights," 29 Oct. 2019, [sysdig.com/blog/sysdig-2019-container-usage-report/](https://sysdig.com/blog/sysdig-2019-container-usage-report/). Accessed 10 Jun. 2020.

# THE CONTAINER MARKET (according to Sysdig)

2020/21 - 2 million Containers  
(a subset of customer containers)



**Fig. 3.** Container runtimes from: "REPORT.2021 Container Security And Usage Report," Jan 2021, [https://dig.sysdig.com/c/pf-2021-container-security-and-usage-report?x=u\\_WFRi](https://dig.sysdig.com/c/pf-2021-container-security-and-usage-report?x=u_WFRi). Accessed 21 Jan. 2021.



**WHAT  
EXACTLY  
IS  
DOCKER?**



# WHAT DO WE ACTUALLY NEED/WANT?

- An isolated environment where a user/application can operate, sharing the host system's OS/kernel without interfering with the operation of another isolated environment on the same system (a container)
- A way to define a container (an image format)
- A way to build an image of a container
- A way to manage container images
- A way to distribute/share container images
- A way to create a container environment
- A way to launch/run a container (a container runtime)
- A way to manage the lifecycle of container instances

# DOCKER, THE WHOLE PACKAGE

DOCKER ENGINE

DOCKER IMAGE FORMAT

Dockerfile `docker build`

`docker images`

`docker rm`

`docker push`

`docker pull`

DOCKER HUB

`docker run`

`docker stop`

`docker ps`



# BREAKING UP THE MONOLITH

## OCI IMAGE FORMAT

- Docker V2 Image Spec

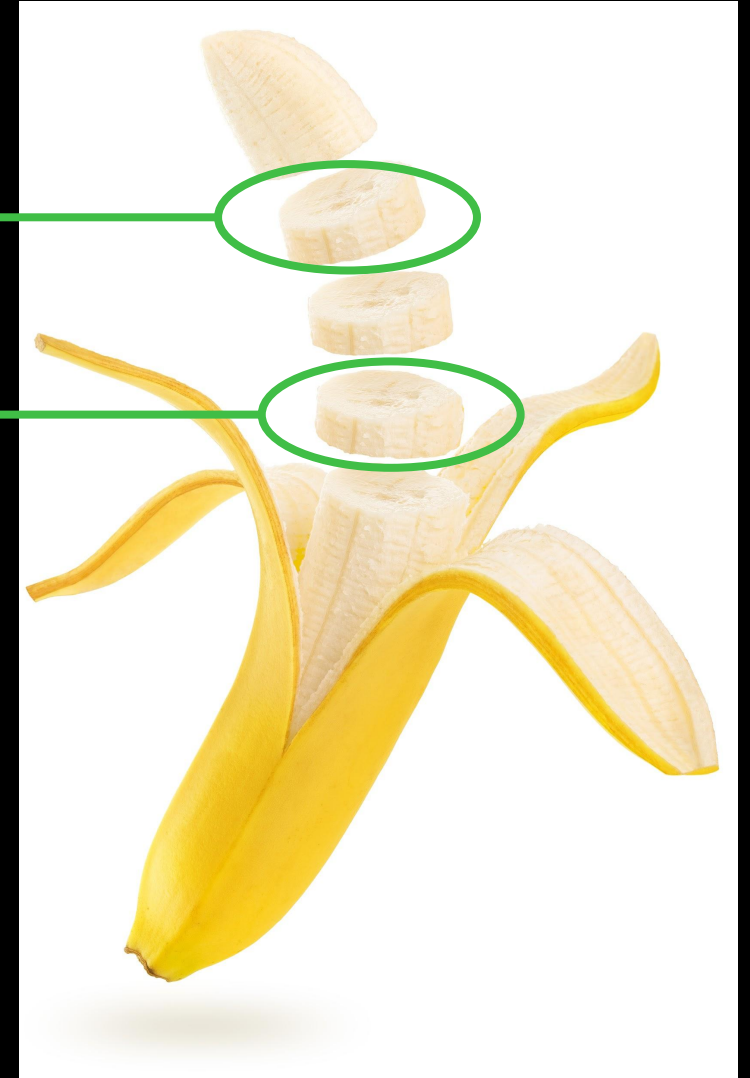
## OCI CONTAINER RUNTIME

- runC (*which used to be libcontainer... which was written by Docker*)

**OTHERS** - containerd, ~~kt~~, cri-o, Kata, etc...

<https://lwn.net/Articles/741897/>

<https://www.ianlewis.org/en/container-runtimes-part-1-introduction-container-r>



**WHAT  
EXACTLY  
IS A  
CONTAINER?**





# CONTAINER COMPONENTS

## TARBALL OF A FILESYSTEM

## LINUX FEATURES

- namespaces
- cgroups
- Union File systems

```
docker-desktop:~# lsns
```

	NS	TYPE	NPROCS	PID	USER	COMMAND
4026532297	mnt		13	13436	999	postgres
4026532298	uts		13	13436	999	postgres
4026532299	ipc		13	13436	999	postgres
4026532300	pid		13	13436	999	postgres
4026532302	net		13	13436	999	postgres

```
~ ▷ docker stats
```

CONTAINER	CPU %	MEM USAGE / LIMIT
d99745e33562	0.01%	480KiB / 1GiB
9094a1844f8e	4.16%	1.338GiB / 1.944GiB
fcf20c230c2c	0.19%	19.41MiB / 1.944GiB

*Mix these together to create and run a container! Voila!*

<https://docs.docker.com/get-started/overview/>

# FILESYSTEM DETAILS

```
~ ▶ docker info
```

```
...
```

```
Operating System: Docker Desktop
```

```
OSType: linux
```

```
Architecture: x86_64
```

```
CPUs: 8
```

```
Total Memory: 1.944GiB
```

```
Name: docker-desktop
```

```
ID: 2POK:GJEZ:EHHW:WDRH:PYOW:PQ6C:LYAB:XLOH:DYSW:4SSN:A3JR:NXUF
```

```
Docker Root Dir: /var/lib/docker
```

```
Debug Mode: true
```

```
File Descriptors: 67
```

```
Goroutines: 76
```

```
...
```



*NOTE: On OSX, containers will actually be running in a tiny Linux VM (use screen)*

```
screen ~/Library/Containers/com.docker.docker/Data/vms/0/tty
```

# FILESYSTEM DETAILS

~ ▷ docker images

REPOSITORY

mjmckay-app-docker.jfrog.io/my-image

docker.bintray.io/jfrog/artifactory-jcr

TAG

latest

7.5.7

IMAGE ID

62165ddeceb6

8b3066e25260

~ ▷ docker inspect 8b3066e25260

[

{

"Id": "sha256:8b3066e252609e484b032c583dada4ebd6f59b6b5de0a2f597f91b5ed4bcf117",

...

"GraphDriver": {

"Data": {

"LowerDir": "/var/lib/docker/overlay2/b01599ceea2761004b4f6a0a0d3d5c368dc40c8f20808434de4ed312029b1ff/diff:/var/lib/docker/overlay2/47dbb7eff56c58762e84b943a98bd1b558b5800b000b98bc6a07bfae53c1d79e/diff:/var/lib/docker/overlay2/2f5763dd07792eb22869fd9118a80d2170eafe6936d78bc73dbc3dc600e

...

"MergedDir": "/var/lib/docker/overlay2/a0bbe2014fe5a7befe1eaaca401a3d2ac54340e7513e719cffc4383722af9406/merged",

"UpperDir": "/var/lib/docker/overlay2/a0bbe2014fe5a7befe1eaaca401a3d2ac54340e7513e719cffc4383722af9406/diff",

"WorkDir": "/var/lib/docker/overlay2/a0bbe2014fe5a7befe1eaaca401a3d2ac54340e7513e719cffc4383722af9406/work"

# FILESYSTEM DETAILS

~ ▶ docker ps -a

CONTAINER ID	IMAGE	COMMAND
d99745e33562	mjmckay-app-docker.jfrog.io/my-image:latest	"/bin/sh -c 'tail -f..."
9094a1844f8e	docker.bintray.io/jfrog/artifactory-jcr:7.5.7	"/entrypoint-artifac..."
fcf20c230c2c	docker.bintray.io/postgres:9.6.11	"docker-entrypoint.s..."

docker-desktop:~# ls /var/lib/docker/

builder	containers	overlay2	swarm	volumes
buildkit	image	plugins	tmp	
containerd	network	runtimes	trust	

docker-desktop:~# ls /var/lib/docker/containers/

9094a1844f8e398845a6ae8f44c1cd9b8ffa21101133a6042ec741faf1ff9b0d  
d99745e335621a1ed138fa1812d7fc83d9c5e337a159f92efd70ed7ed46df4b0  
fcf20c230c2cc706a82bc16a6b9e39ee8a8d82b6508bd03cfd80d1ea2715106c

~ ▶ docker rm my\_image\_name

~ ▶ docker prune

~ ▶ docker run -d --memory=1g mjmckay-app-docker.jfrog.io/my-image:latest --rm





# CONTAINER GOTCHAS

# CONTAINER GOTCHAS

## - RUNNING AS ROOT



# CONTAINER GOTCHAS

- NO CONSTRAINTS



# CONTAINER GOTCHAS

## - NEVER UPDATING





# CONTAINER GOTCHAS

## - JAVA/JVM GOTCHAS



# CONTAINER GOTCHAS

## - IMAGE BLOAT



# MANAGING YOUR IMAGES

## - REMOTE BY DEFAULT



**START FREE:**

[http://bit.ly/JFokus\\_Free\\_DevOps\\_Tools](http://bit.ly/JFokus_Free_DevOps_Tools)



**foojay.io**

foojay — a place for friends of OpenJDK

<https://dzone.com/refcardz/getting-started-with-container-registries>

# THE SWAG - DON'T FORGET!

**WIN**  
A JFROG SHIRT!



<http://bit.ly/JFokusJFrog2021>







# Q & A

## THANK YOU!

Melissa McKay



@melissajmckay



linkedin.com/in/melissajmckay