OPITZ CONSULTING

■■■ Überraschend mehr Möglichkeiten!

Docker 101

Getting Started with Docker

Philipp Perez

Organizational Matters

- English
- Wifi
- VM
- Pizza, Beer, Fritz-Kola

OTTO

kibana

NGINX

Agenda

1 Docker

2 Docker Terminology

3 Hello, world

4 Docker Subcommands

5 Dockerfile

Agenda

6 Golang Docker Image

7 Python Docker Image

8 Node Docker Image

9 Docker Registry

10 Docker on Windows & Mac

Docker

Docker

- Open Source Containerization engine or container platform
- Automates packaging, shipping and deployment
- Apps are presented as lightweight and portable containers
- Containers will run everywhere
- Written in Go(lang)

Docker

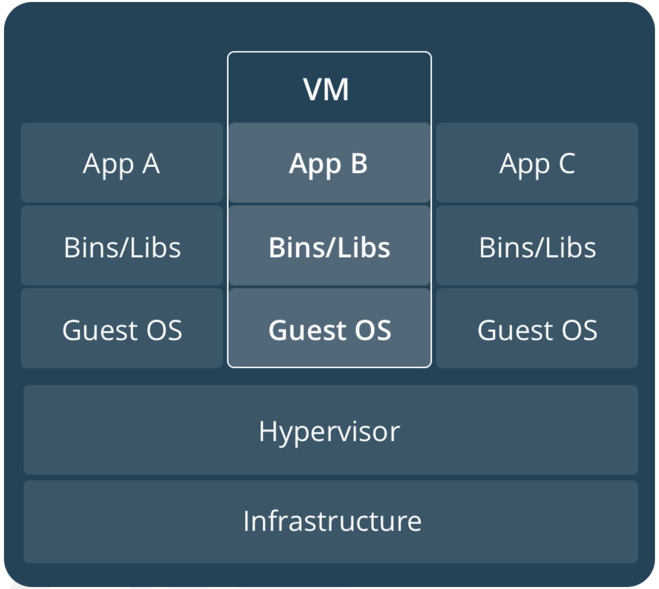
Virtualization vs. Containerization

- Hardware-level virtualization
- Bundle complete OS + libraries, settings, apps
- OS is resource intensive
- Takes up a lot of space (3 – X GB)
- Slow to boot
- Fully isolated

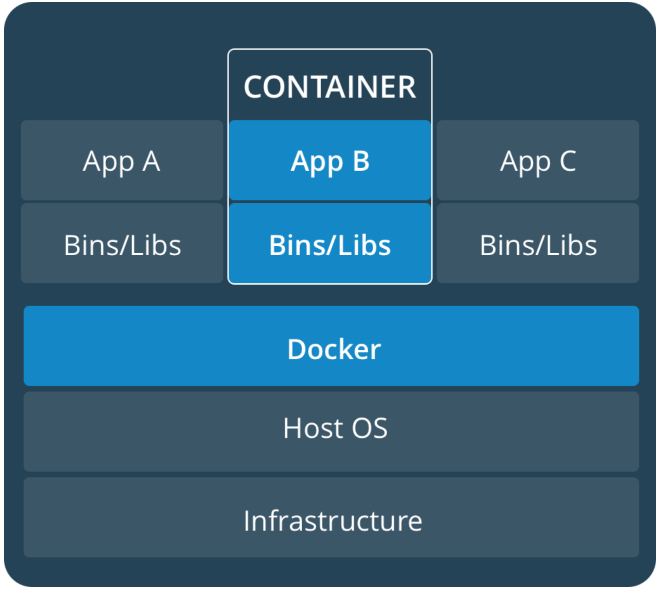
- Operating system virtualization
- Bundle libraries, settings, apps
- More portable + efficient
- Takes up less space (100 - 800 MB)
- Start almost instantly
- Process-level isolation

Docker

Virtualization vs. Containerization



Virtual Machines



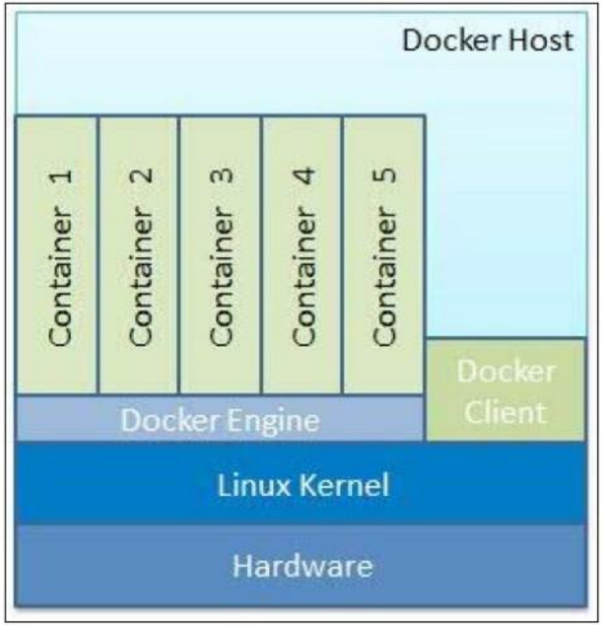
Containers

https://docs.docker.com/get-started/

Docker

Virtualization vs. Containerization

- Containers share a single kernel
- Run natively on the host’s machine kernel
- Better performance
- Each container in separate process



eBook: Learning Docker by Jeeva S. Chelladhurai, p. 3

Docker

Docker Index

- Publicly available repository of images

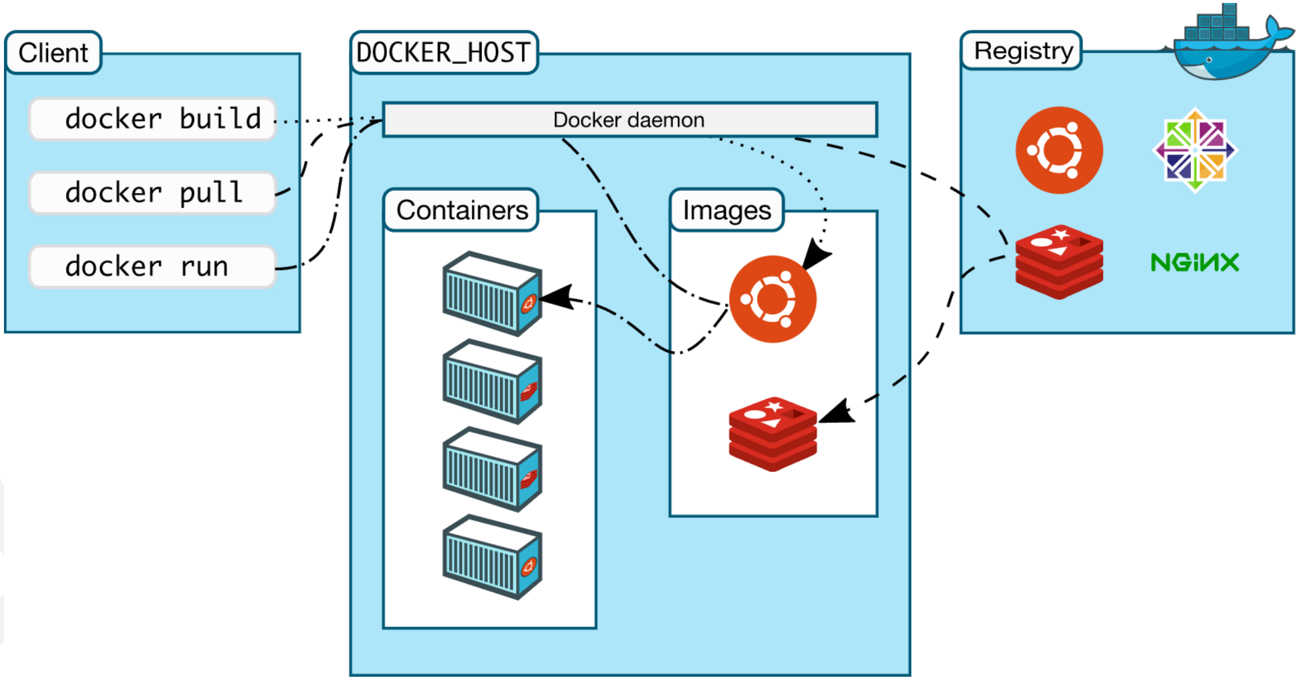
  → http://index.docker.io / http://hub.docker.com

- Official images and third-party images,

  e.g. Ubuntu, CentOS, NGINX

Docker

Docker Architecture



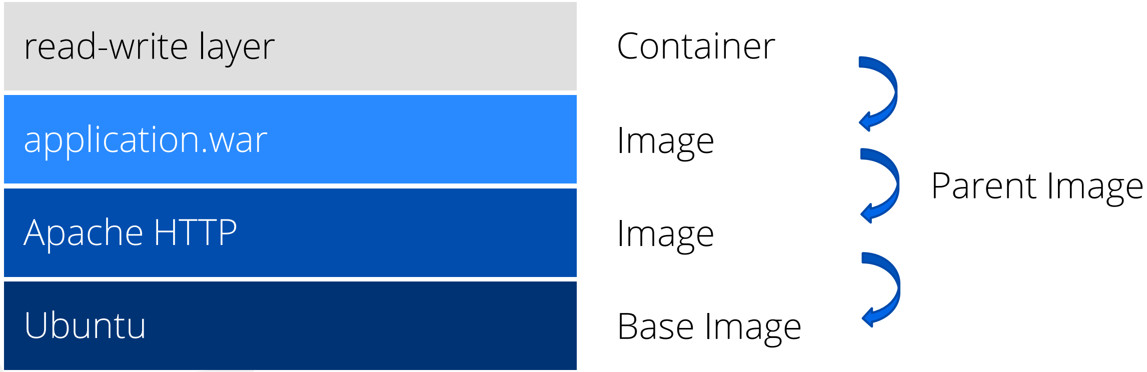
https://docs.docker.com/engine/docker-overview/#docker-architecture

Docker Terminology

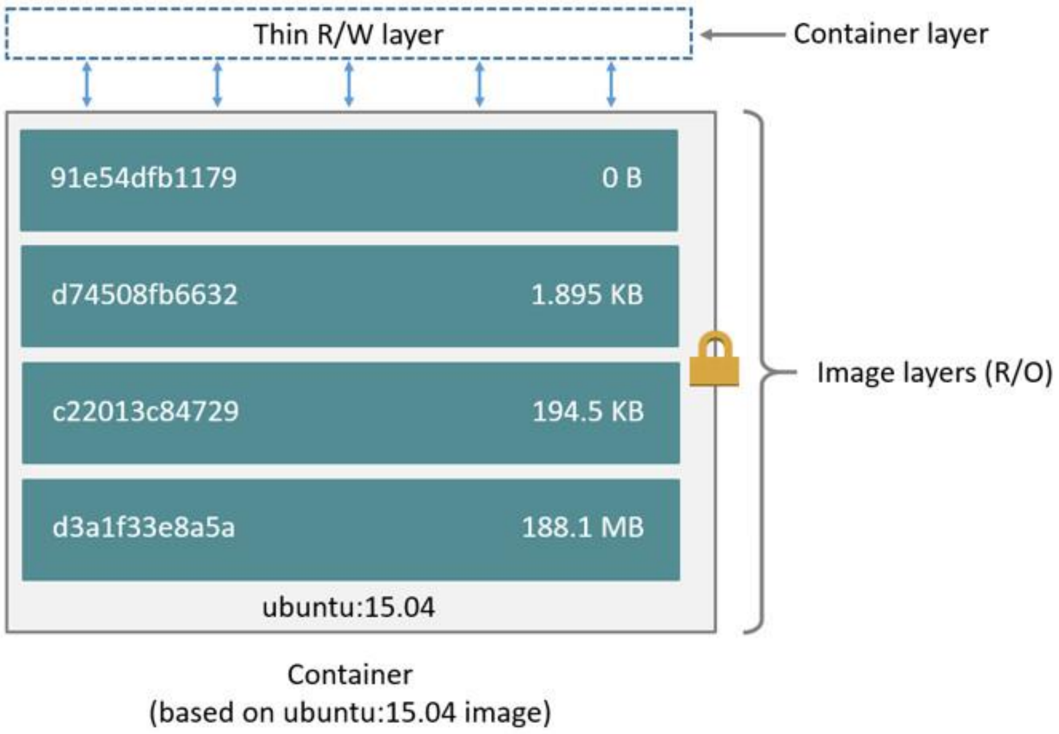
Docker Terminology

- Docker Image:
  - Read-only template with instructions for creating a Docker container
  - Collection of all files that make up a software application
  - Consists of multiple layers
- Docker Container:
  - Runnable instance of an image
  - You can create, start, stop, pause and delete containers
  - Multiple containers of the same image

Docker Terminology



Docker Terminology



Base Images:

- Alpine
- Debian
- Busybox
- Ubuntu
- CentOS
- Etc.

Docker Terminology

- Docker images have a name + optional tags

  redis:3.2.11

- Identification of Docker containers and images:
  - Unique ID
  - 64 Hex digit identifier (SHA-256 hash)
  - (Random) Name

Docker Terminology

Paradigms & Conventions

- Process with PID 1 determines container's lifetime

  → only one process / application per container

- Log to STDOUT / STDERR

- Use environment variables for configuration

3

docker run hello-world

Docker Subcommands

Docker Subcommands

- docker pull <image_id>
- docker images
- docker run < image_id>
- docker ps
- docker ps -a

- Detach: Ctrl + p, Ctrl + q
- docker attach <container_id>
- docker stop < container_id>
- docker rm <container_id>

Docker Subcommands

Alpine Linux

- Famous Linux distribution
- Much smaller, ~ 5 MB
- Designed to run in RAM
- Leads to smaller images

→ recommended to use

Docker Subcommands

Alpine Linux

- Available packages in Alpine:
  - top
  - ps
  - wget
  - grep
  - ifconfig
  - vi

- Install additional packages:

```
apk add --no-cache git
```

Docker Subcommands

Alpine Linux Exercise

- docker pull alpine
- docker images
- docker run alpine
- docker ps
- docker ps -a

- docker run -it alpine sh
  - top
  - ps
  - wget
  - grep
  - ifconfig
  - vi
  - Detach: Ctrl + p, Ctrl + q
- docker ps
- docker attach <CONTAINER_ID>
- docker stop <CONTAINER_ID>

Docker Subcommands

Docker Run

-d	Detached mode (daemon)
-e <key>=<value>	Environment variable
--name <container_name>	Name
-p <port>:<port>	Expose ports
-it	Interactive mode
-v <host_src>:<container_dst>	Mount volume
<image_id> <command>	

Docker Subcommands

Docker Run

```
docker run \
    -it \
    --name alp \
    -p 8080:8080
    -e PIPELINE=dev
    -v /home/user/workspace:/workspace \
    alpine
```

Let’s try it out!

Docker Subcommands

Tracking Changes in Docker

- docker diff <container_id>
- docker commit < container_id> <new_image_name>

- docker images
- docker history <image_id>

Docker Subcommands

Tracking Changes in Docker

1. Start an Alpine Linux container in interactive mode.
   - Add a file.
   - Delete a file.
2. Exit the container.
3. View the difference.
4. Commit the changes of your container.
5. View the history of the newly created image.

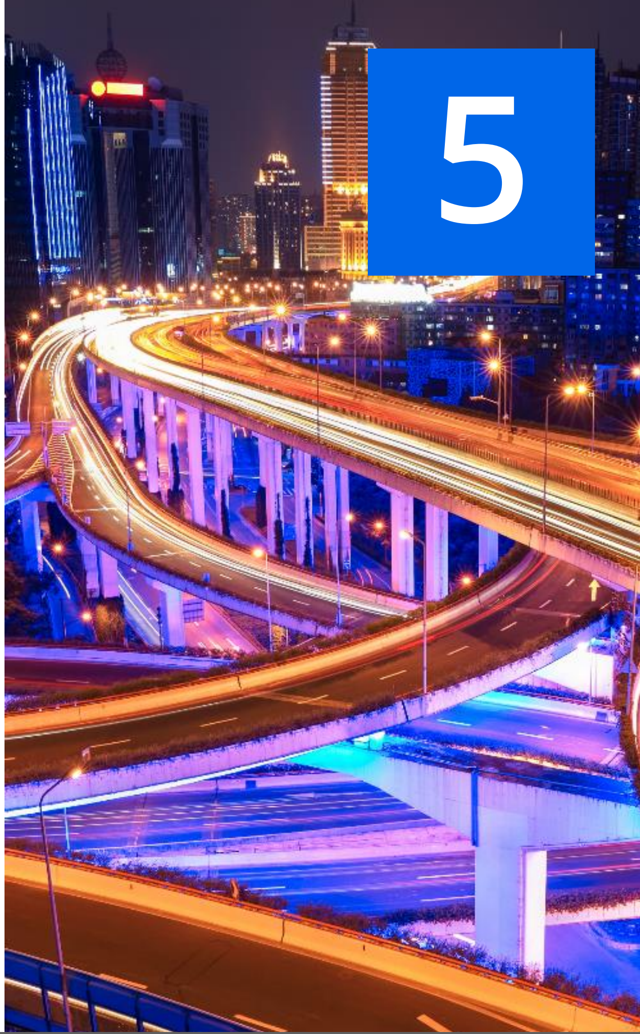
Docker Subcommands

Tracking Changes in Docker

→ Few steps to create an image from a container.

⚠ Use this method only for testing purposes!

Dockerfile



Dockerfile

- Text file containing all commands for building Docker images
- Specific format + set of instructions
- Docker can build images automatically by reading Dockerfiles

  docker build –t <image_name> .

Dockerfile

- Build context + Dockerfile are send to Docker Engine
- Docker daemon runs instructions one-by-one
- Each instruction creates a new layer / image
- Build cache

Dockerfile

Dockerfile Instructions

- FROM:
  - Must be the first instruction
  - Selects the parent / base image
  - FROM <image>[:<tag>]
- ADD or COPY:
  - Copies files from Docker host to FS of new image
  - ADD can handle tar files + URLs
  - COPY <src> <dst>

Dockerfile

Dockerfile Instructions

- RUN:
  - Executes any kind of command, e.g. *apt-get update*
  - RUN <command>
- ENTRYPOINT or CMD:
  - Executes any kind of command
  - Similar to RUN, but executed when container is launched
  - Specifies executable of container (PID 1)

Dockerfile

Dockerfile Instructions

- ENTRYPOINT or CMD:
  - Only last ENTRYPOINT / CMD will have an effect
  - Difference:
    - ENTRYPOINT can't be overridden by subcommand *docker run*
    - *docker run* subcommand arguments are passed as additional args
  - ENTRYPOINT ["<exec>", "<arg-1>“, ..., "<arg-n>"]
  - CMD ["<exec>", "<arg-1>“, ..., "<arg-n>"]

Dockerfile
Dockerfile Instructions

- ENV:
  - Sets environment variables
  - ENV <key>=<value>
- WORKDIR:
  - Changes current working directory
  - Relative or absolute path
  - WORKDIR <dirpath>

Dockerfile

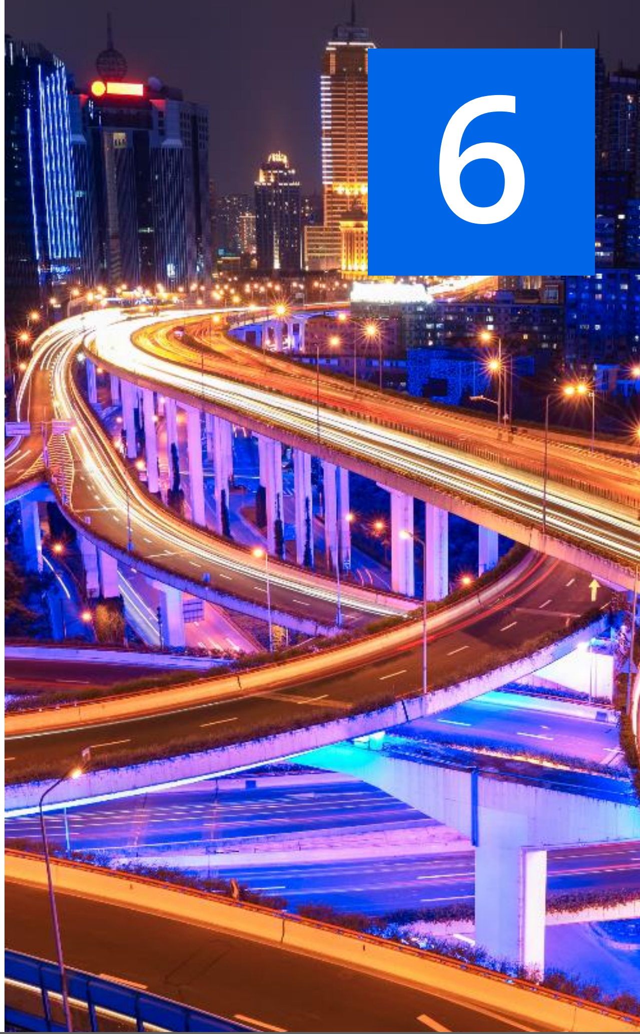
Dockerfile Instructions

- EXPOSE:
  - Informs Docker that this containers listens on the specified network port
  - For communicating between container and host machine
  - Does not actually publish the port → documentation
  - EXPOSE <port>
- See Dockerfile reference for more

Golang Docker Image

git clone

https://github.com/philenius/dockerCodeKata



Golang Docker Image

Golang Application with web server :8080

- go get
- go build

→ Install the Go package

- Ubuntu as base image
- $GOPATH

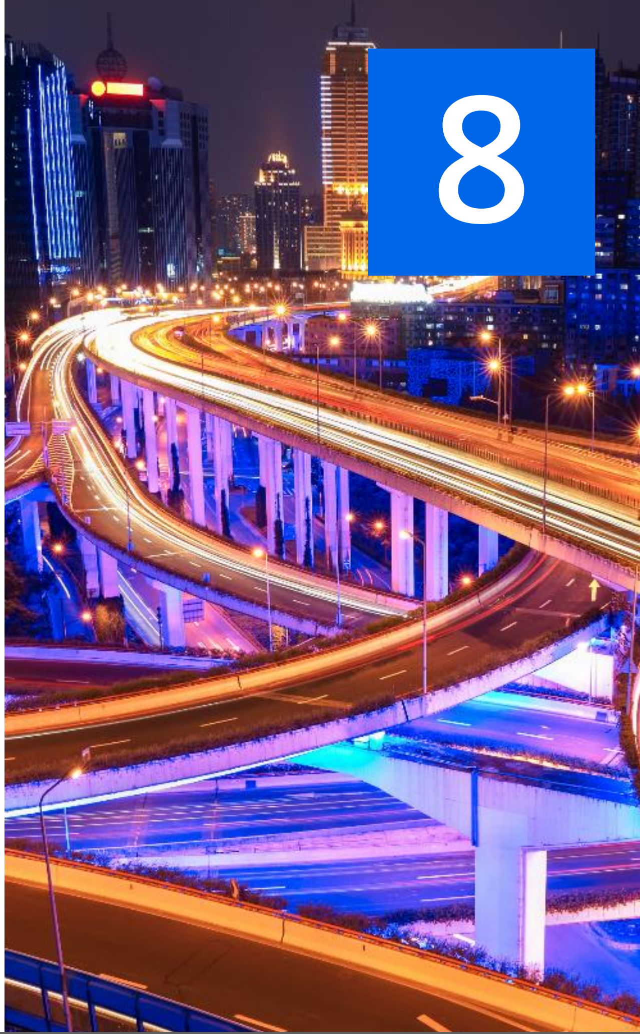
→ Binary

Golang Docker Image

- Two helpful subcommands:
  - docker logs <container_id>
  - docker inspect <container_id>

Python Docker Image

Node Docker Image



8

Docker Registry

9

Docker Registry

- Place where Docker images can be stored
- Sharing Docker images (publicly)
- Registry registers images
- Repositories stores actual Docker images
- Naming convention: <user_id>/<repository_name>
- Public registry: Docker Hub

Docker Registry

- Tag Docker images:

  docker tag <image> <ip>:5000/<repository>

- Push Docker images:

  docker push $\underbrace{\text{<ip>:5000/<repository>}}_{\text{image}}$

Docker Registry

- Wifi:

  SSID: Meetup

  Password: docker101

- Registry:

  192.168.178.26:5000

- Exercise:

  1. Tag one of your images.
  2. Push it to our registry.
  3. Ask a neighbor for his image.
  4. Pull his image.
  5. Run his image.

Docker on Windows & Mac

10

Docker on Windows & Mac

- Docker is built on top of the Linux kernel
- Can only be directly run on Linux distributions
- Windows & Mac systems don’t meet the requirements

→ Docker Toolbox

Docker on Windows & Mac

Docker Toolbox

- Lightweight Linux VM with help of adapters
- Enables the Docker Engine to run on Windows & Mac
- Oracle Virtual Box

Docker on Windows & Mac

Docker for Windows & Mac

- Native Docker Engine without virtualization
- Docker for Windows:
  - Hyper-V
  - Windows 10 with 1607 Anniversary Update Build 14393
- Docker for Mac:
  - xhyve
  - OS X 10.10.3 Yosemite

Q & A

Do more.

Philipp Perez

Big Data Software Engineer

Weltenburger Straße 4
81677 München

Philipp.Perez@opitz-consulting.com
+49 89 680098-1440

WWW.OPITZ-CONSULTING.COM

@OC_WIRE

OPITZCONSULTING

opitzconsulting

opitz-consulting-bcb8-1009116