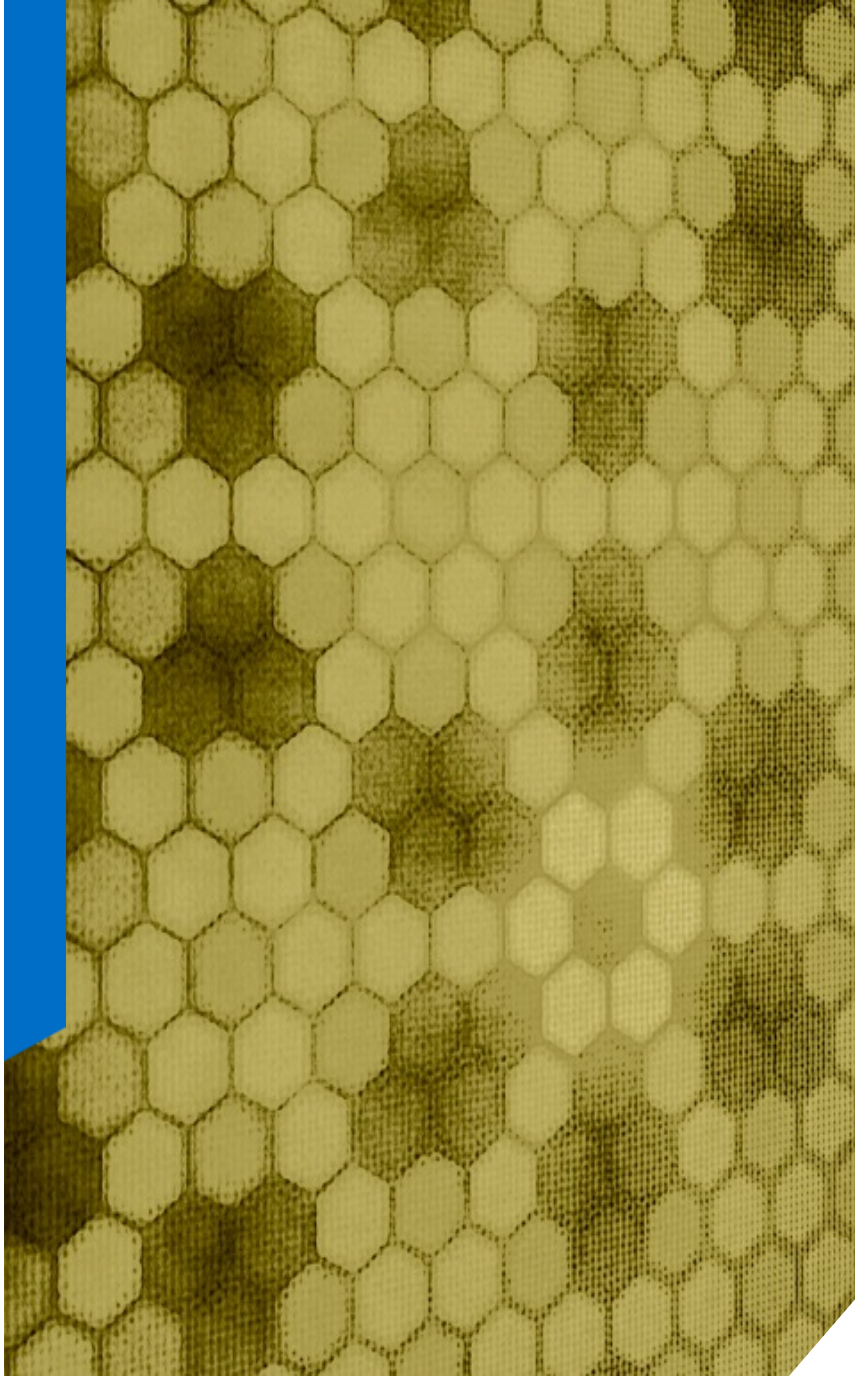


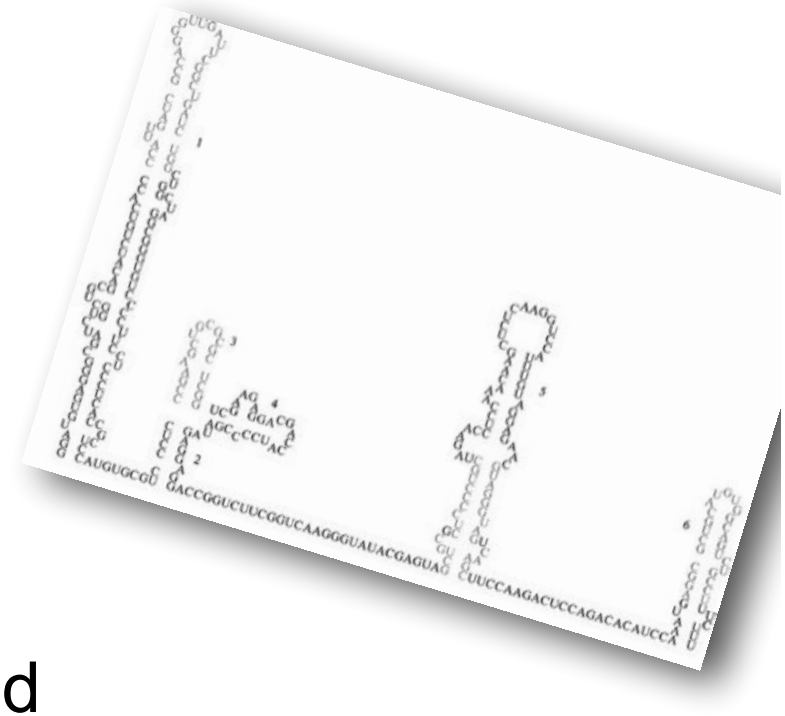
adesso business. people. technology.

Dr. Halil-Cem Gürsoy
@hgutwit

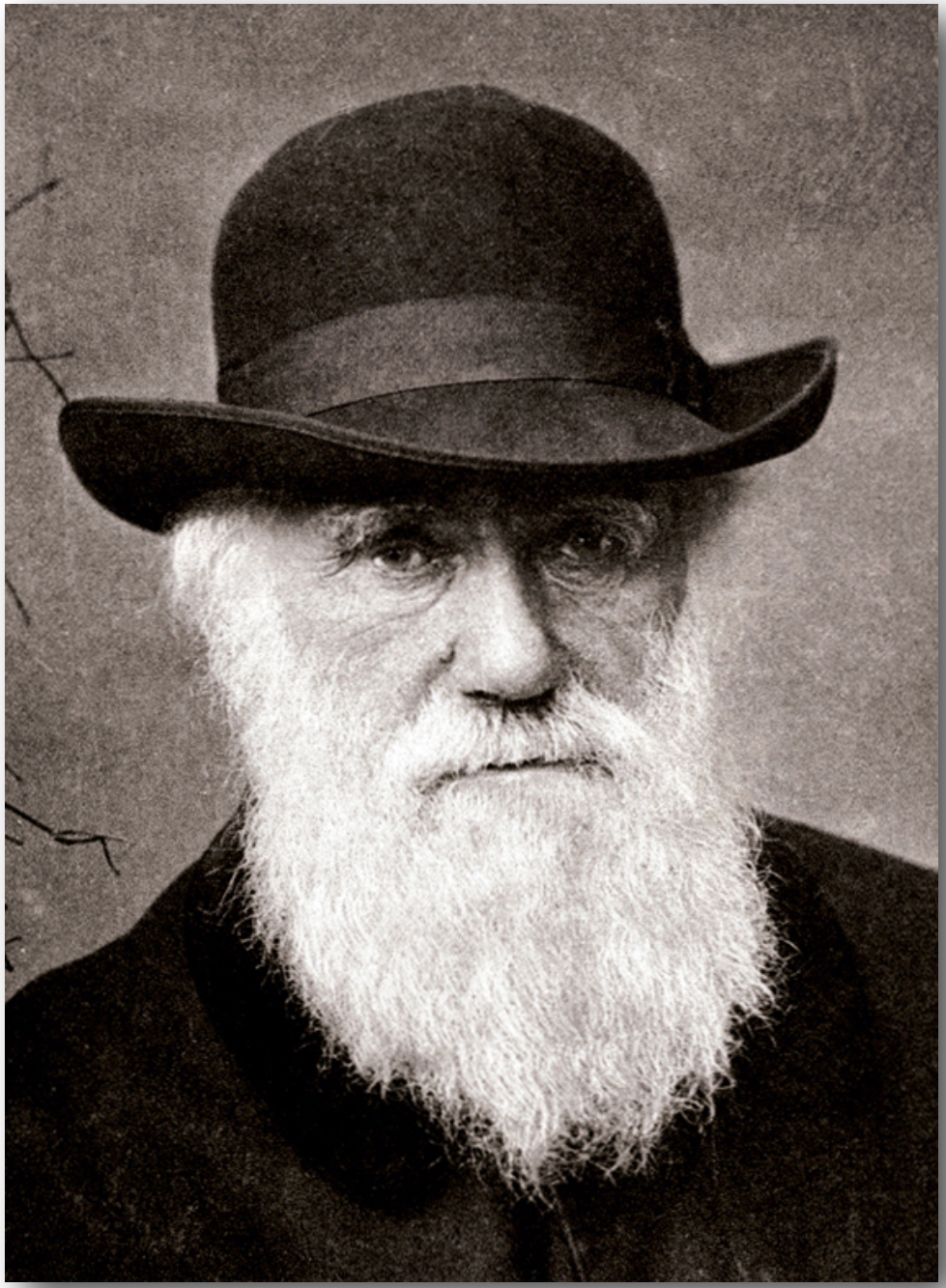
adesso AG

Continuous Delivery in der Praxis

Über mich



- Principal Architect @ adesso AG
- seit 15 Jahre Software-Entwicklung
  - > davor in wissenschaftlichem Umfeld
- Verteilte Enterprise-Systeme
- Persistenz / Build & Deployment



„Our highest priority is to satisfy the customer through early and ***continuous delivery*** *of* ***valuable software.“***

Agile Manifesto Principles

Continuous Integration

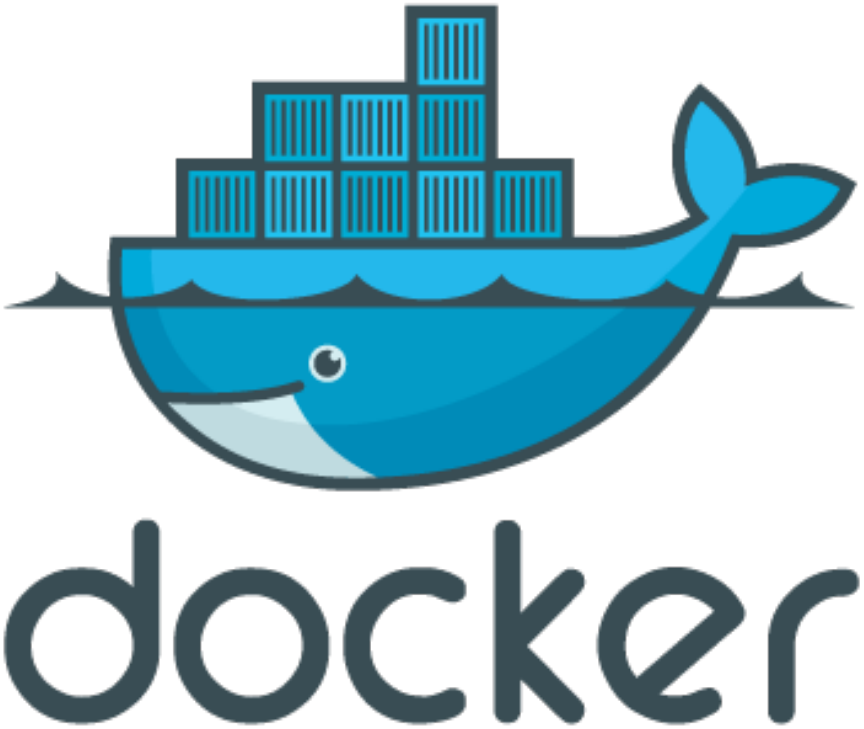
- Integration von Modulen
- Vielleicht auch Integrationstests
- Fokus ist Entwicklung, nicht Delivery

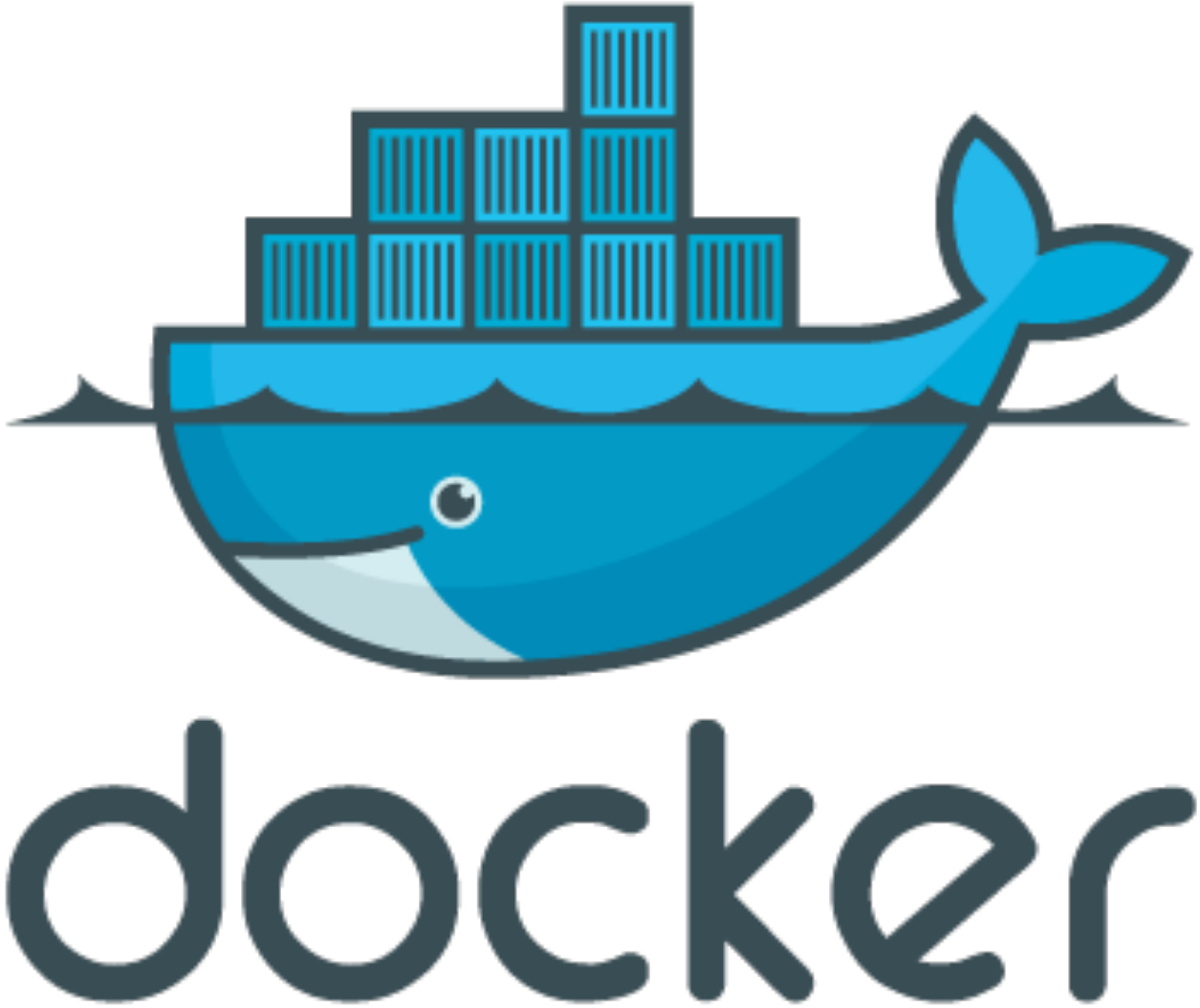
- Keine Infrastruktur, Life-Tests…
- Viele manuelle Schritte


https://www.flickr.com/photos/striatic/2191404675/

Continuous Delivery

- Kontinuierliche Builds
- Artefakte werden **vollständig** durchgetestet
  - Incl. automatisierte UAT, CAPT usw.
  - Incl. Server-Setup und Infrastruktur
  - Incl. Deployments
- Keine manuellen Eingriffe
- Jeder Build wird potentiell produktiv gesetzt


docker

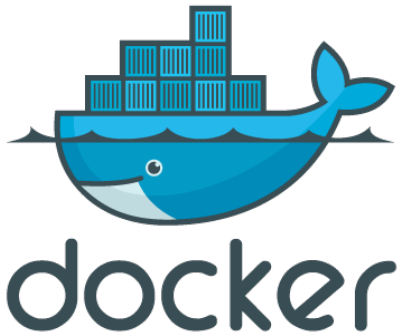

docker


https://www.flickr.com/photos/jpmartineau/501718334/

Warum Docker ?

- ‚Self containing' Container
  - Alles drin was der Prozess benötigt
  - Kein WAR & EAR-Delivery
  - Kein Application Server Setup beim Kunden
  - ‚Run & forget'
- Erleichtert das Leben zw. Dev und Ops

Was ist Docker?

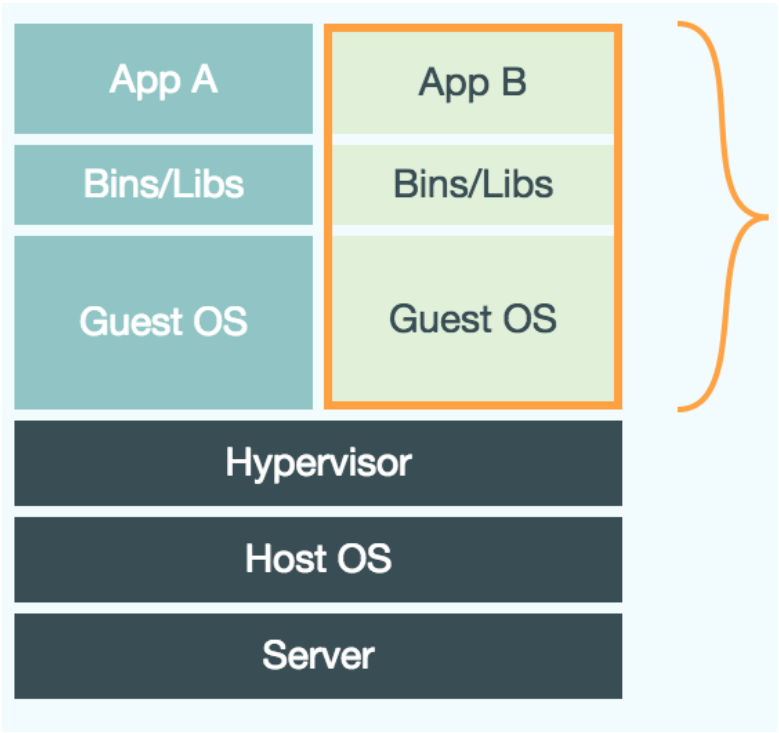


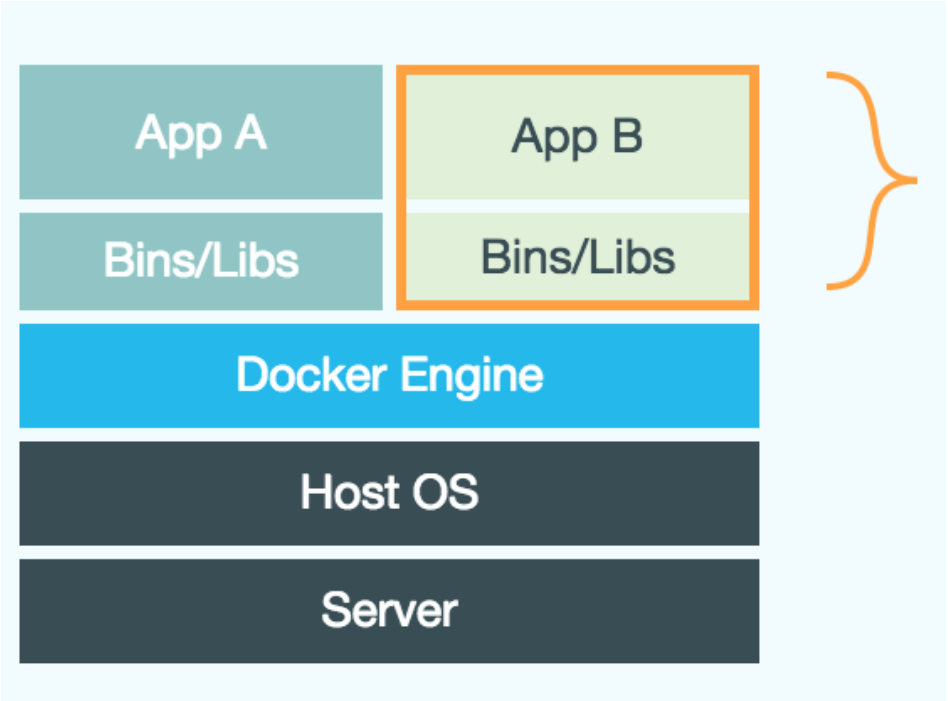
- Implementiert in ‚Go‘
- Nutzt viele Linux Kernel Features:
  - *namespaces* für alle Ressourcen
  - Control groups (*cgroups*)
  - Union File Systeme (AUFS, btrfs, vfs…)
  - LXC, libcontainer

VM vs. Container

- *cgroups* & *namespaces* & *libcontainer*
  - **Isolierte Prozesse**
  - sieht nur alles innerhalb seines *namespace*
  - innerhalb des Containers wie in einer ‚VM‘
- Kernel Sharing
  - es muss kein vollständiges OS gebootet werden

VM vs.Container



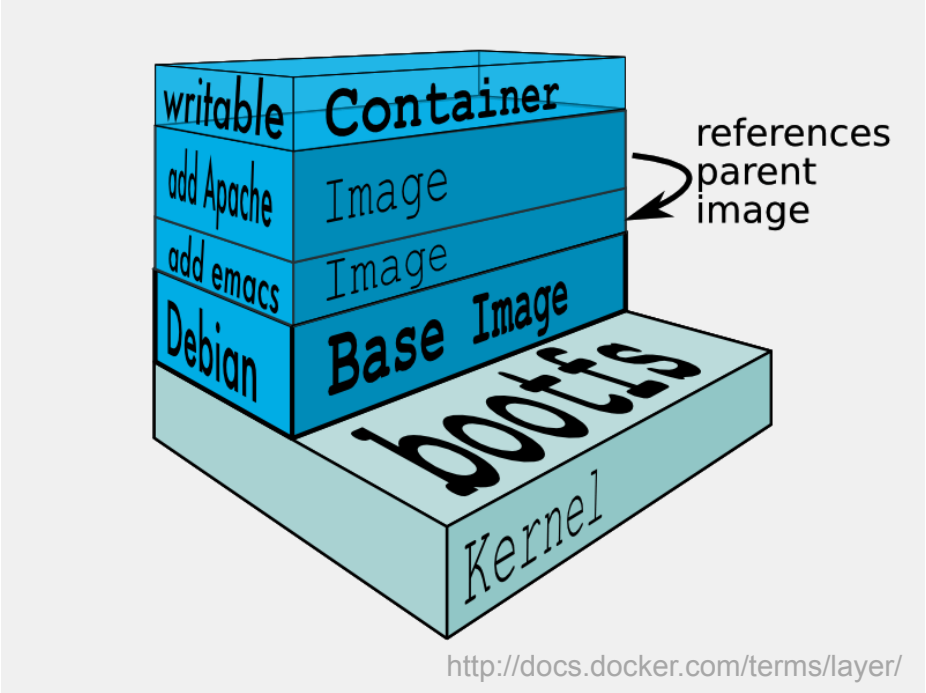


Quelle: http://docs.docker.com

https://www.flickr.com/photos/ashleyswilkes/3863767875

File System

- Unified Filesystem - AUFS
  - > Änderungen werden ‚geschichtet‘
  - > Analog einem Versionierungssystem



http://docs.docker.com/terms/layer/

AUFS

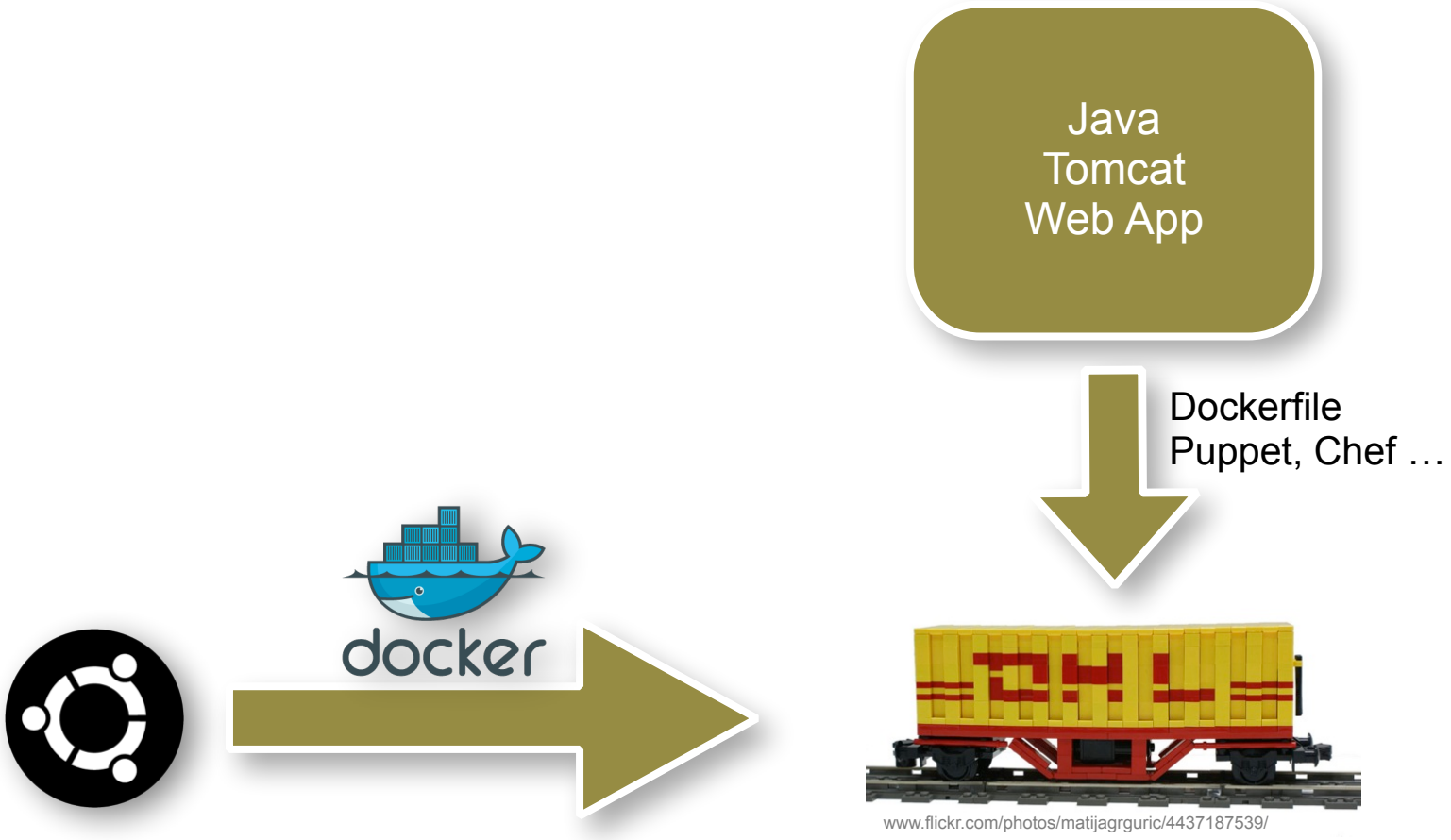
```
└─3588c05c7d24 Virtual Size: 510.5 MB Tags: cdwrkshp/java7-tc7:latest
  └─7cd0471cb71e Virtual Size: 510.5 MB
    ├─54dc3c2d593c Virtual Size: 517 MB
    │ └─094cc7c59c79 Virtual Size: 523.1 MB
    │   └─688a08b73f44 Virtual Size: 523.1 MB
    │     └─52e46ca36cf0 Virtual Size: 523.1 MB Tags: angularjs-springmvc:1.1.0-42
    ├─7cc63bc98af7 Virtual Size: 517 MB
    │ └─8bf219561a14 Virtual Size: 523.1 MB
    │   └─ebe7ec7d34bb Virtual Size: 523.1 MB
    │     └─23205cc10892 Virtual Size: 523.1 MB Tags: angularjs-springmvc:1.1.0-41
    ├─d420a23f980b Virtual Size: 517 MB
    │ └─4afb0caceae6 Virtual Size: 523.1 MB
    │   └─aed914cc5e2c Virtual Size: 523.1 MB
    │     └─d17e5a80a367 Virtual Size: 523.1 MB Tags: angularjs-springmvc:1.1.0-40
```

Dockerfiles

```
FROM ubuntu:precise
MAINTAINER Halil-Cem Guersoy hcguersoy@gmail.com
RUN apt-get install -y wget curl
RUN apt-get install -y openjdk-7-jdk
RUN apt-get install -y tomcat7
COPY ./server.yaml /etc/tomcat7/server.yaml
EXPOSE 8080
```

```
$ docker build –t myrepo:5000/mytc:0.1 .
$ docker inspect myrepo:5000/mytc:0.1
$ docker push myrepo:5000/mytc:0.1
$ docker run –i –t myrepo:5000/mytc:0.1 /bin/bash
$ docker ps -a
```

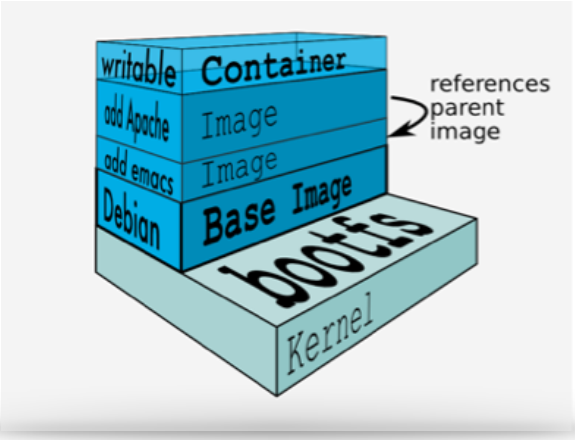
Application Images



Application „Container“

https://www.flickr.com/photos/matijagrguric/4437187539

Layering im Filesystem



http://docs.docker.com/terms/layer/

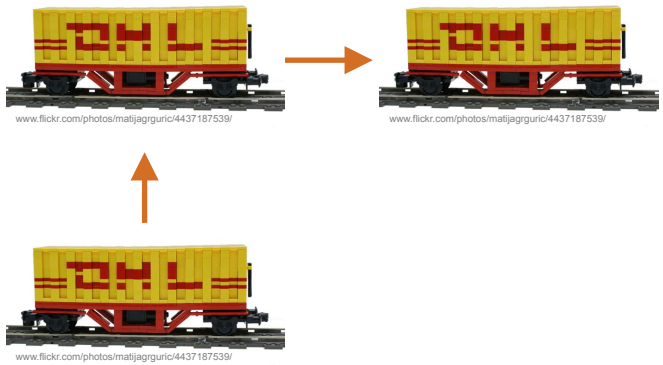
- Basis Images
- Vererbung / Layering File System
- App Image enthält zusätzlich nur Änderungen
  - > Konfiguration & Applikation
- **Jede Version wird nur 1x gebaut!**

Container verbinden

- Mehrere Container benötigt
  - > Datenbank, AppServer, WebServer, LB…
- Container mit Links verbinden
- Oder ‚Registry'-Systeme
  - > etcd, Consul usw.
- Scheduler
  - > Docker Swarm, CoreOS

Container Links

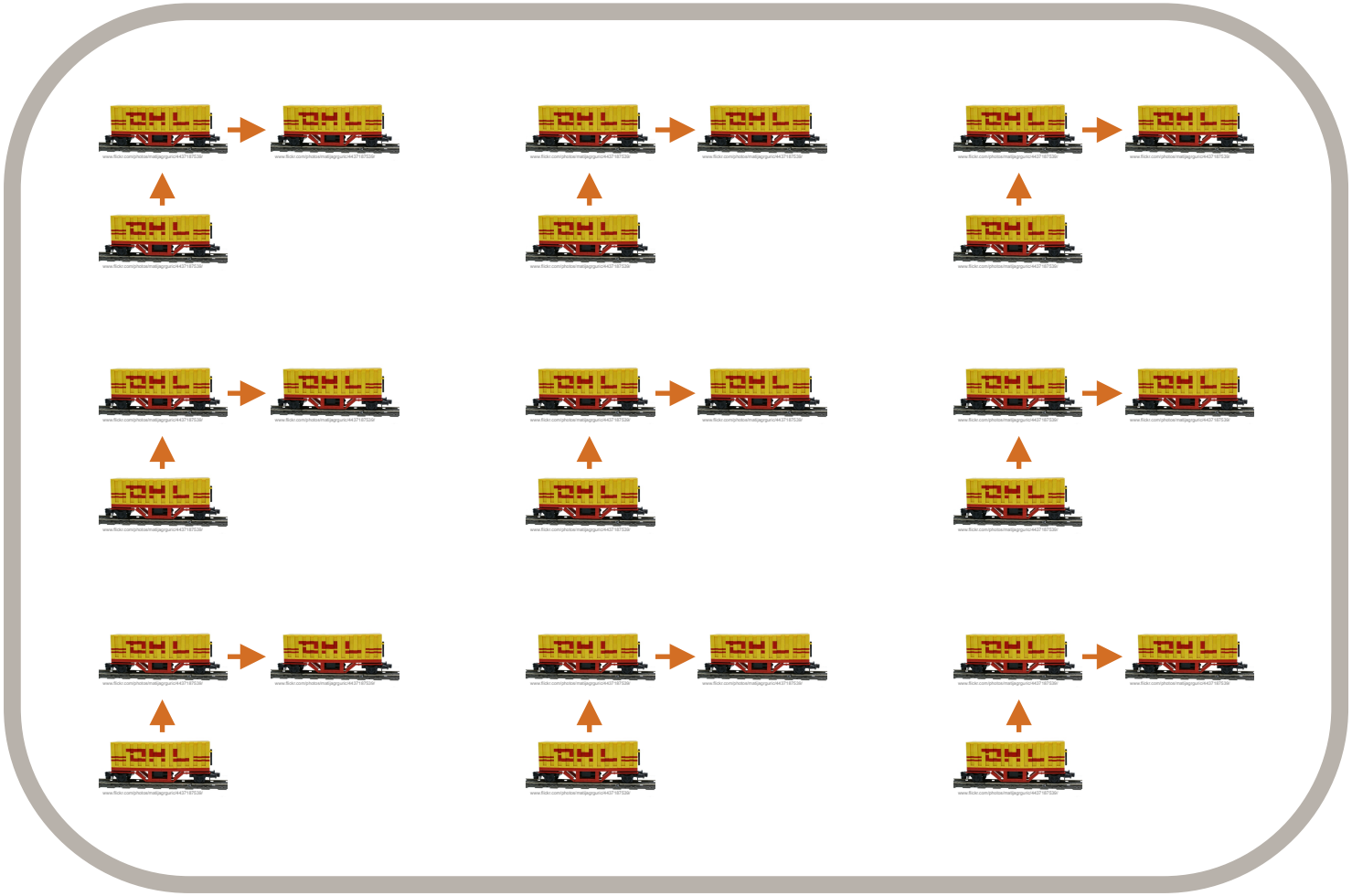
```
docker run -d -P --name db42
docker run -d -P --name app42 --link db42:db

--link [name]:[alias]
```



Container ansprechen

- Port Mapping auf Host
- Default Mapping oder vorgeben
- Expose im Dockerfile
- „Isolierte“ Ports

adesso


DHL

Auswirkungen auf Architektur?

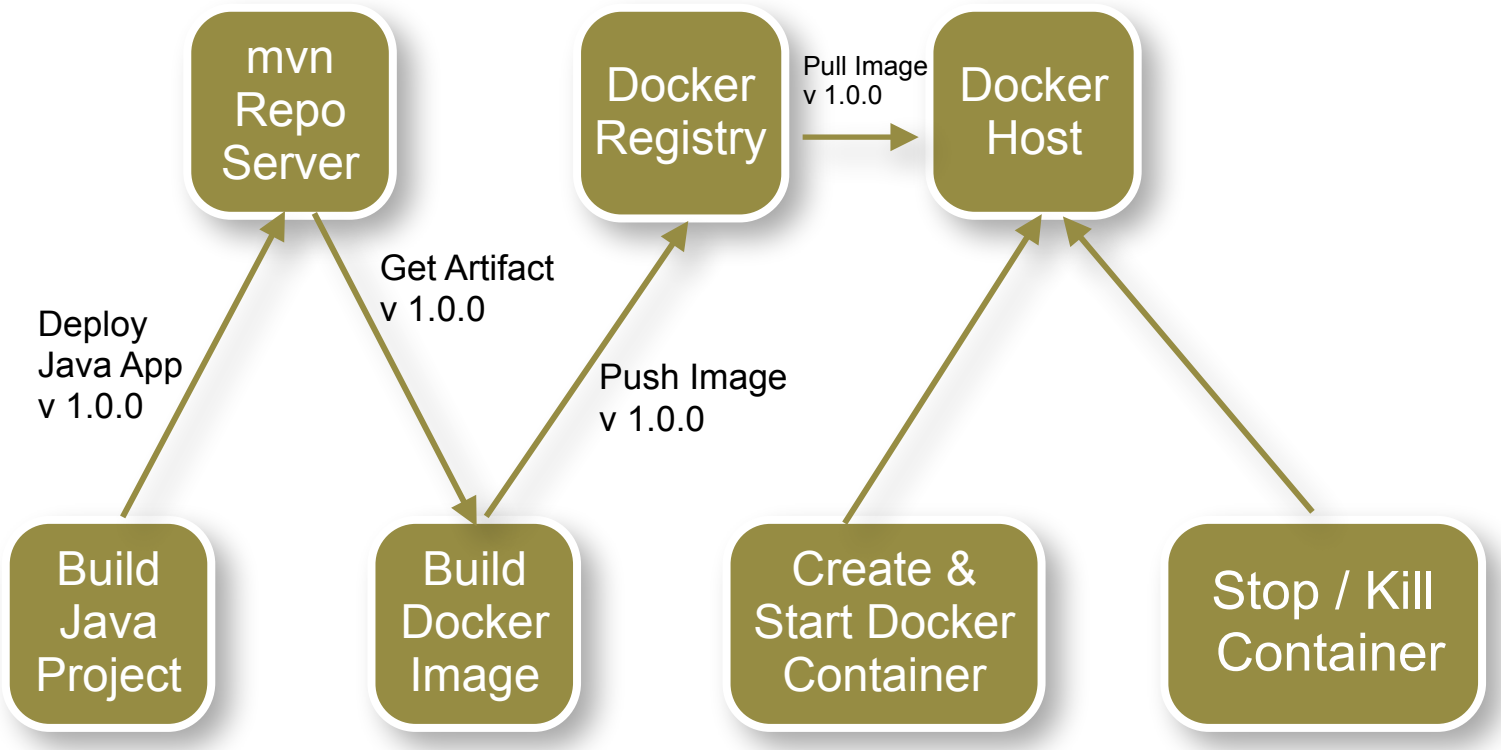
- Weg von großen Monolithen
- Hin zu kleineren Einheiten
- **Microservices**
- Herausforderungen
  - Orchestrierung
  - Resilience

Infrastructure as Code

- Server für Container muss bereit gestellt werden
- CI-Systeme, Docker, … müssen installiert werden
- Puppet, Chef, Ansible…

```
service { $servicename :
    name       => $servicename,
    ensure     => running,
    enable     => true,
    require    => [File["/etc/init/$servicename.conf"]],
    subscribe => File[$config_file],
}
```

CD + Docker Workflow - supertrivial

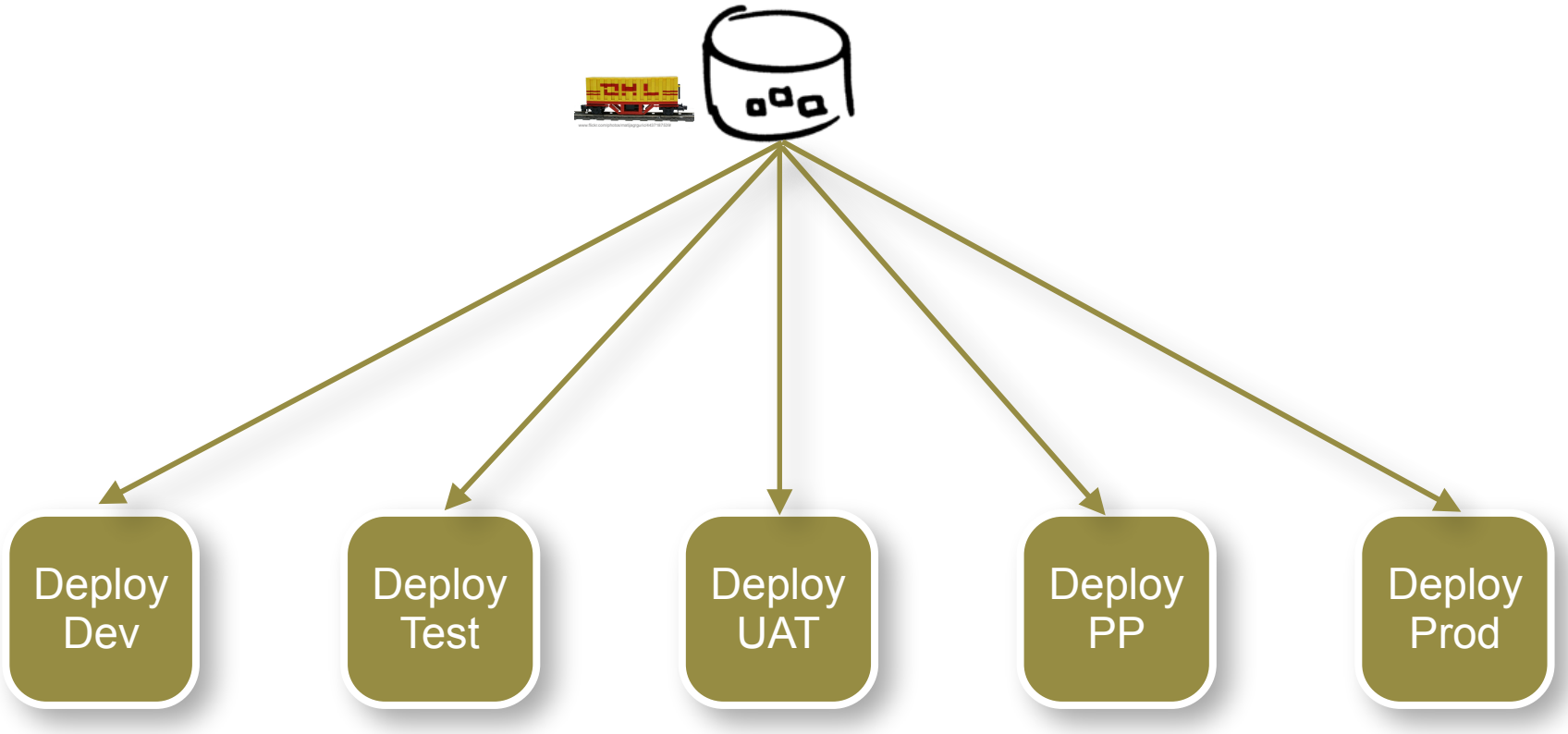


adesso

https://www.flickr.com/photos/jdhancock/4071590385/

https://www.flickr.com/photos/jstar/2338025074

Promote!



Jenkins

- „nur“ ein C. Integration-Server
- CD Funktionalität über Plugins

- Plugins harmonieren nicht immer :-(

Docker mit Jenkins

- TODOs:
  - Images bauen, pullen, pushen
  - Container anlegen und starten
  - Container stoppen / löschen etc.

- *Docker build step plugin*
  - leider nicht ganz ausgereift

REST to the Rescue

- Docker bietet REST API
- Alle notwendigen Aktionen aufrufbar
- Integration in Jenkins via Shell / Scripting
- Client APIs für fast alle wichtigen Sprachen:
  http://docs.docker.com/reference/api/remote_api_client_libraries/

docker-py

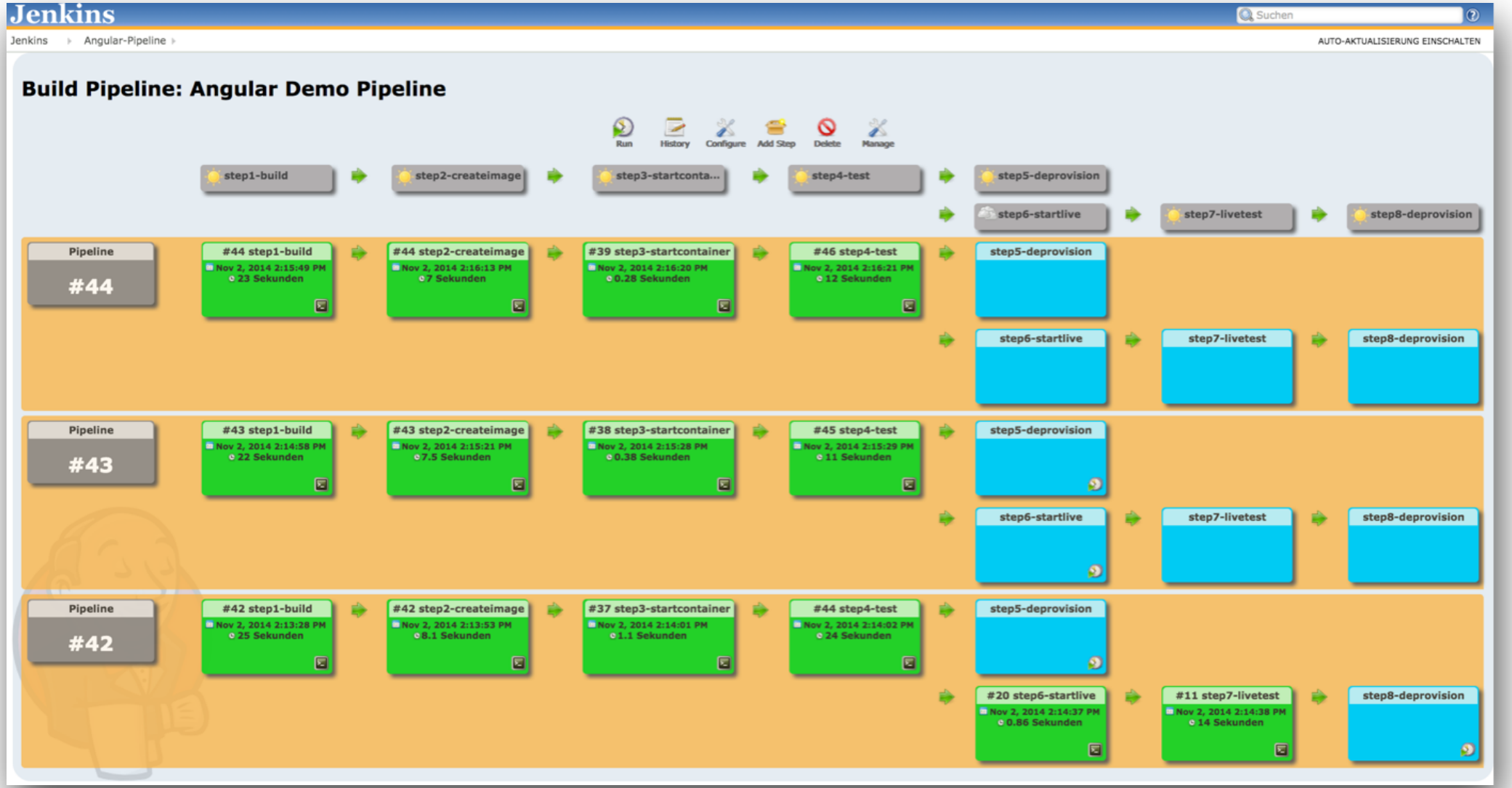
- Kommt von Docker
- Python „Standard“ auf Linux
- Einfache Client API
- Über „Shell Exekution“ in Jenkins aufrufbar

docker-py

```
def create_start_container(repository, image, tag, name, links):
    pull_container(repository, image, tag)
    cont_info = c.create_container(repository+"/"+image+":" + tag, name=name)
    print 'The ID is :', cont_info["Id"]
    if args.verbose > 1:
        print "Container Informationen:", cont_info
    started = c.start(cont_info, publish_all_ports=True, links=links)
    print "Container gestartet ", cont_info["Id"]
    return cont_info;
```

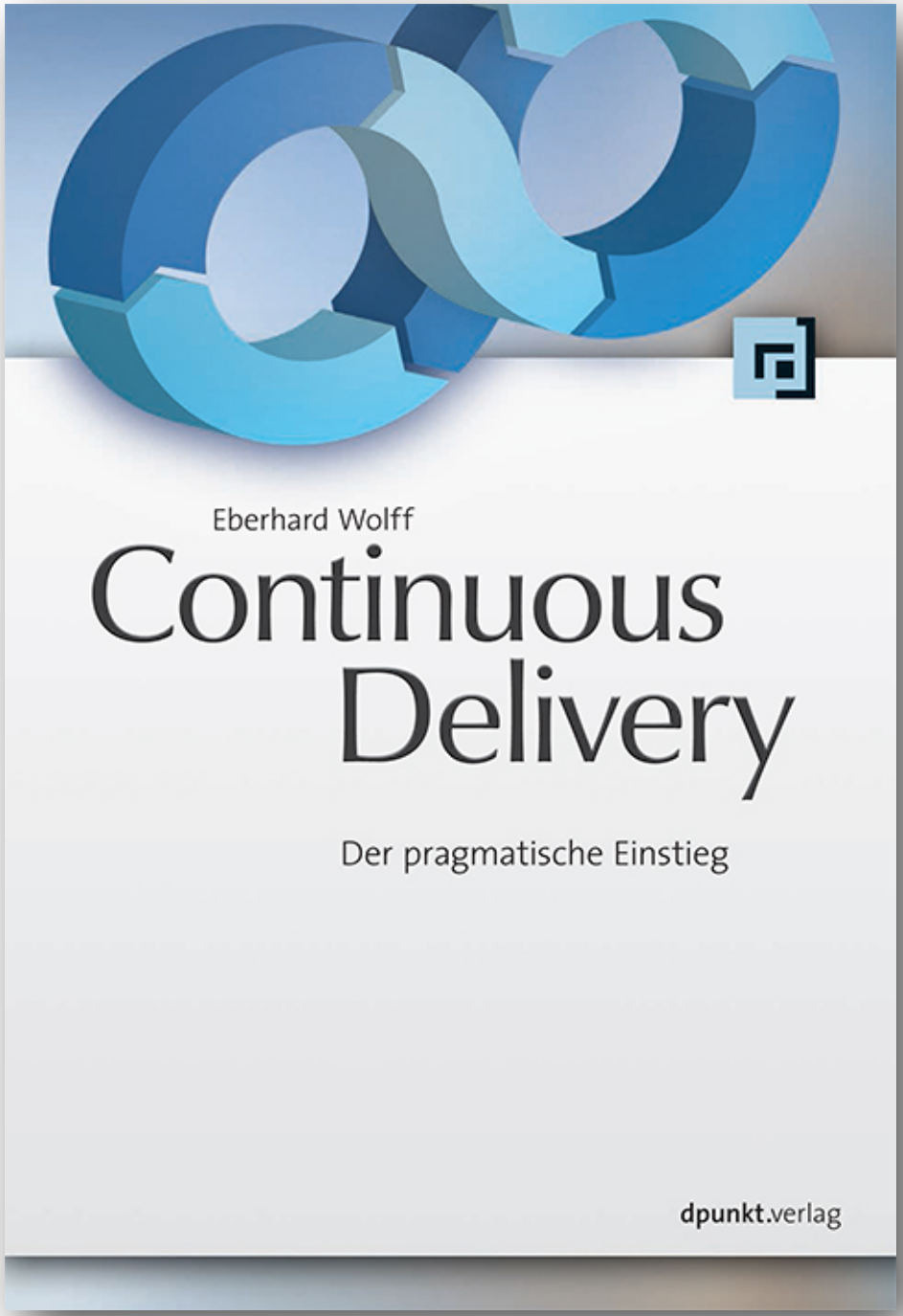
Noch mehr Plugins

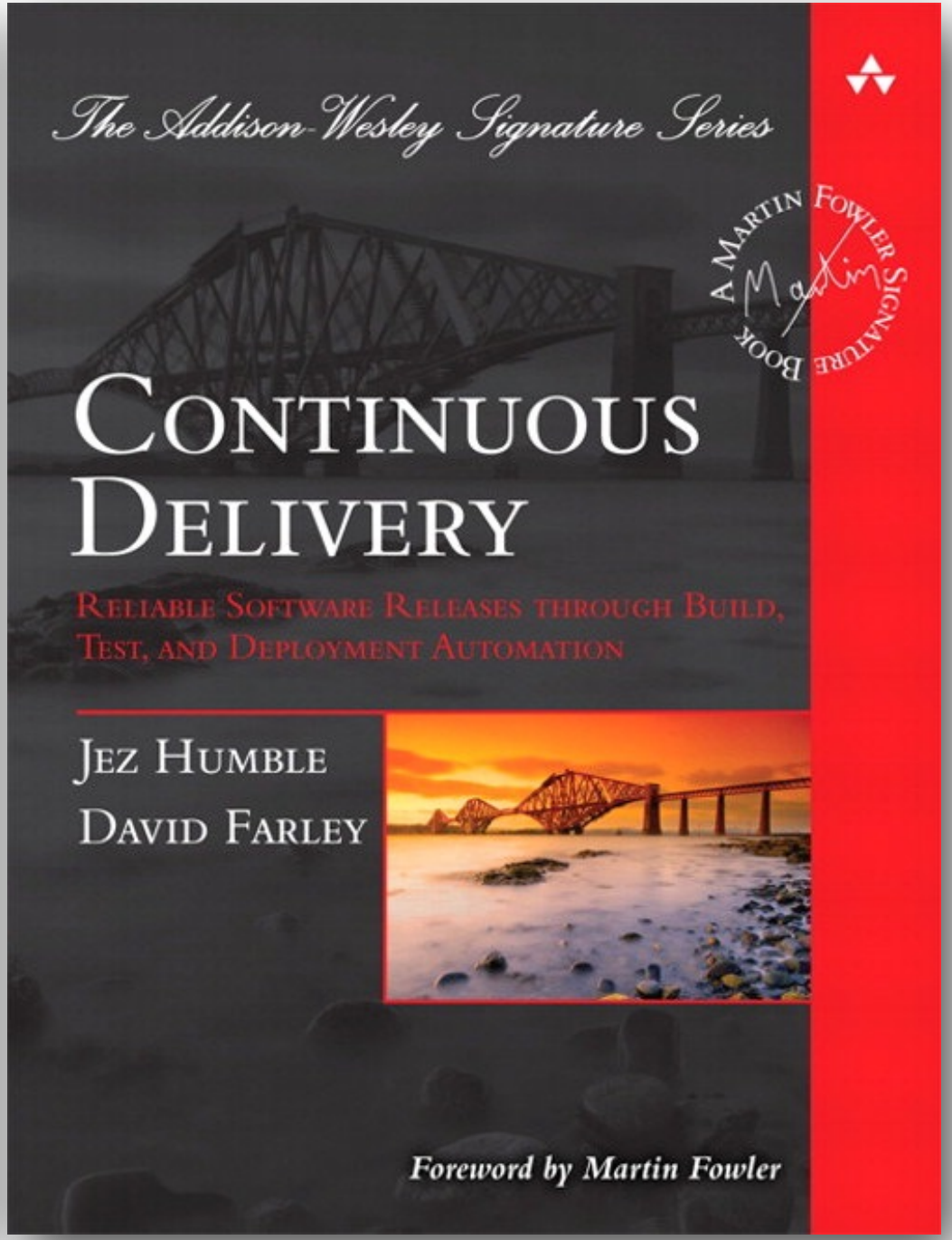
- Parametrizied Trigger Plugin
- Build Pipeline Plugin
- Rebuild Plugin
- Artifact Resolver Plugin
- Artifact Promotion Plugin
- Jenkins Workflow Plugin


Jenkins
Suchen
Jenkins
Angular-Pipeline
AUTO-AKTUALISIERUNG EINSCHALTEN
Build Pipeline: Angular Demo Pipeline
Run
History
Configure
Add Step
Delete
Manage
step1-build
step2-createimage
step3-startconta...
step4-test
step5-deprovision
step6-startlive
step7-livetest
step8-deprovision
Pipeline
#44
#44 step1-build
Nov 2, 2014 2:15:49 PM
23 Sekunden
#44 step2-createimage
Nov 2, 2014 2:16:13 PM
7 Sekunden
#39 step3-startcontainer
Nov 2, 2014 2:16:20 PM
0.28 Sekunden
#46 step4-test
Nov 2, 2014 2:16:21 PM
12 Sekunden
Pipeline
#43
#43 step1-build
Nov 2, 2014 2:14:58 PM
22 Sekunden
#43 step2-createimage
Nov 2, 2014 2:15:21 PM
7.5 Sekunden
#38 step3-startcontainer
Nov 2, 2014 2:15:28 PM
0.38 Sekunden
#45 step4-test
Nov 2, 2014 2:15:29 PM
11 Sekunden
Pipeline
#42
#42 step1-build
Nov 2, 2014 2:13:28 PM
25 Sekunden
#42 step2-createimage
Nov 2, 2014 2:13:53 PM
8.1 Sekunden
#37 step3-startcontainer
Nov 2, 2014 2:14:01 PM
1.1 Sekunden
#44 step4-test
Nov 2, 2014 2:14:02 PM
24 Sekunden
#20 step6-startlive
Nov 2, 2014 2:14:37 PM
0.86 Sekunden
#11 step7-livetest
Nov 2, 2014 2:14:38 PM
14 Sekunden

Jenkins Skalierung

- Docker zum Skalieren von Jenkins
- „Jenkins Docker Plugin“
- „Slave Container“ on demand
- Spezialisierte Build-Container
  - DinD für Image Builds


Eberhard Wolff
Continuous Delivery
Der pragmatische Einstieg
dpunkt.verlag


The Addison-Wesley Signature Series
A Martin Fowler Signature Book
Continuous Delivery
Reliable Software Releases through Build, Test, and Deployment Automation
Jez Humble
David Farley
Foreword by Martin Fowler

http://www.flickr.com/photos/an_untrained_eye/6630719431

halil-cem.guersoy@adesso.de
https://twitter.com/hgutwit

http://www.flickr.com/photos/tcmorgan/7372944070