



Container im Schwarm - Weblogic und Docker  
Swarm

# Bio



- Thorsten Wussow
- 20 Jahre IT davon 16 Jahre mit Oracle
- [thorsten@slix.de](mailto:thorsten@slix.de)
- Twitter: @thwussi



# Agenda



- Einleitung
- Docker 1.12 und der Swarm Mode
- Weblogic 12.2.1.2 im Swarm
- Demo
- Q&A

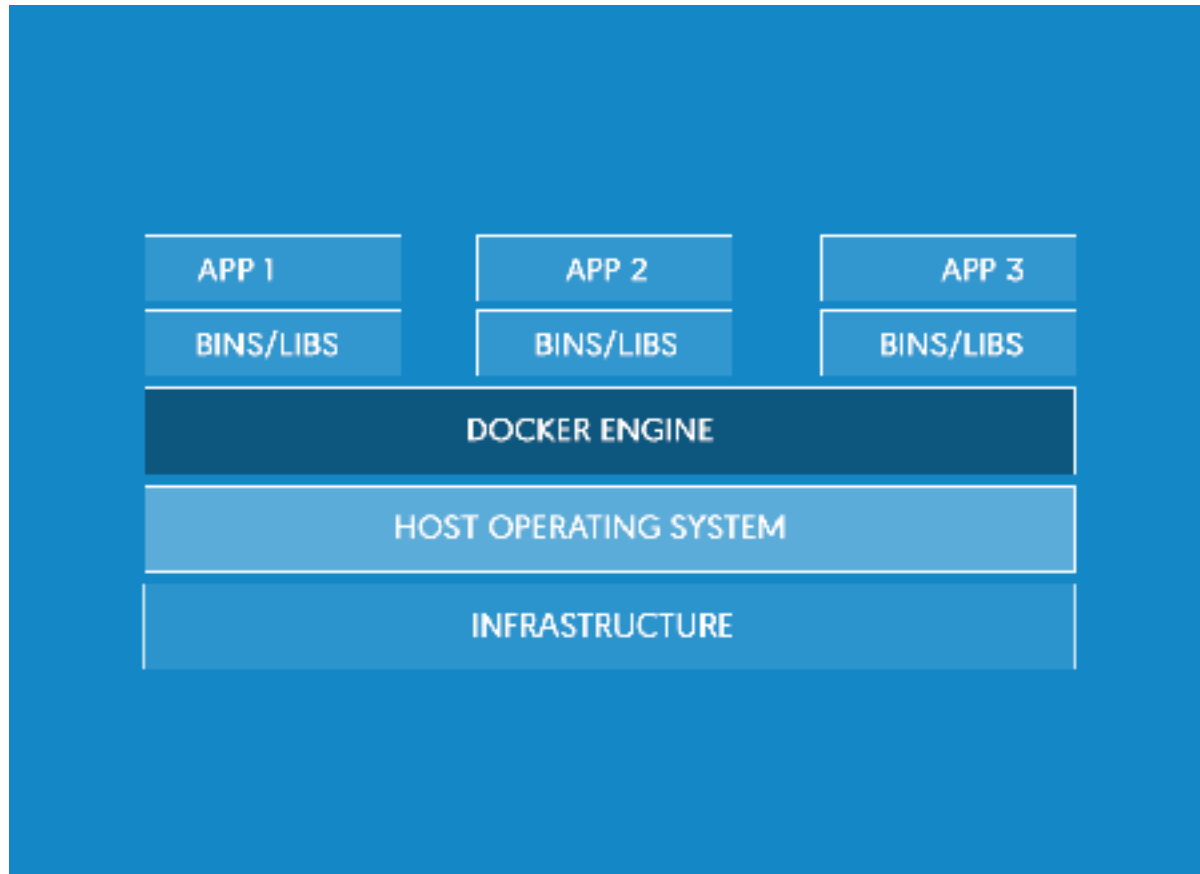
# Einleitung



- Docker ist aktuell in aller Munde und Hip
- Konzipiert für Microservices
- Schnelle Entwicklungszyklen von Docker
- Vortrag setzt Grundwissen von Docker voraus, Schwerpunkt liegt bei Swarm und Vernetzung



# Docker Schema Laufzeitumgebung



# Agenda



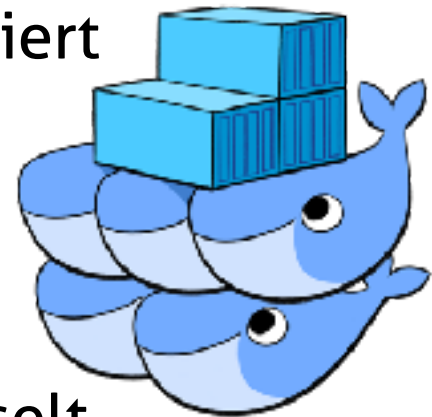
- Einleitung
- Docker 1.12 und der Swarm Mode
- Weblogic 12.2.1.2 im Swarm
- Demo
- Q&A

# Docker 1.12 Swarm-Mode



# Docker Swarm-Mode

- Seit Docker 1.12 mit build-in Orchestration
- Swarm-Mode in docker-engine enthalten
- Neue API-Befehle um einen Swarm aufzubauen
- alle Möglichkeiten zur Orchestrierung integriert
- Integriertes Loadbalancing
- Multihost-Networking
- Internes DNS
- Alle Verbindungen im Swarm sind verschlüsselt





# Docker 1.12 under the hood



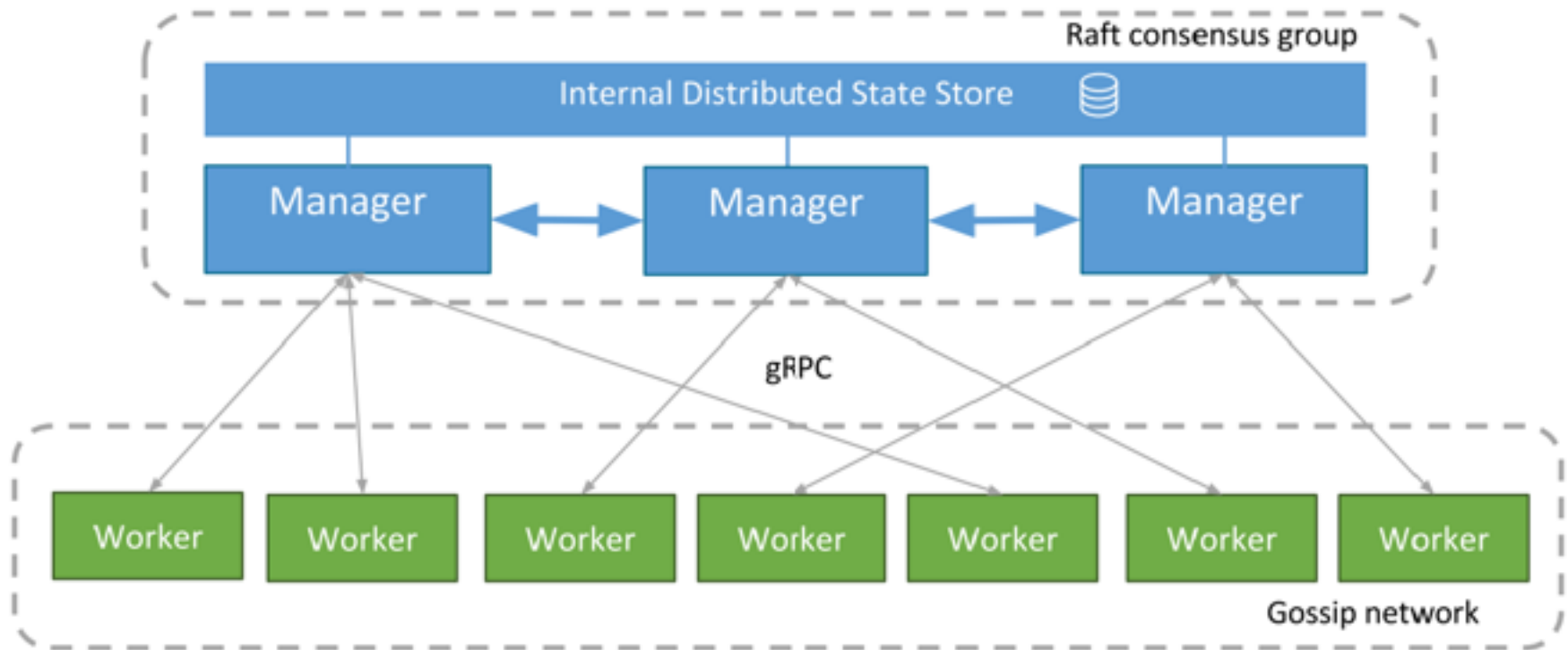
- ManagerNodes arbeiten mit Raft-Consensus-Algorithm
- Damit wird dafür gesorgt, das alle ManagerNodes den gleichen Status des Clusters haben
- Im Grunde eine Log-Replikation zwischen den ManagerNodes
- Gute Erklärung unter <http://thesecretlivesofdata.com/raft/>



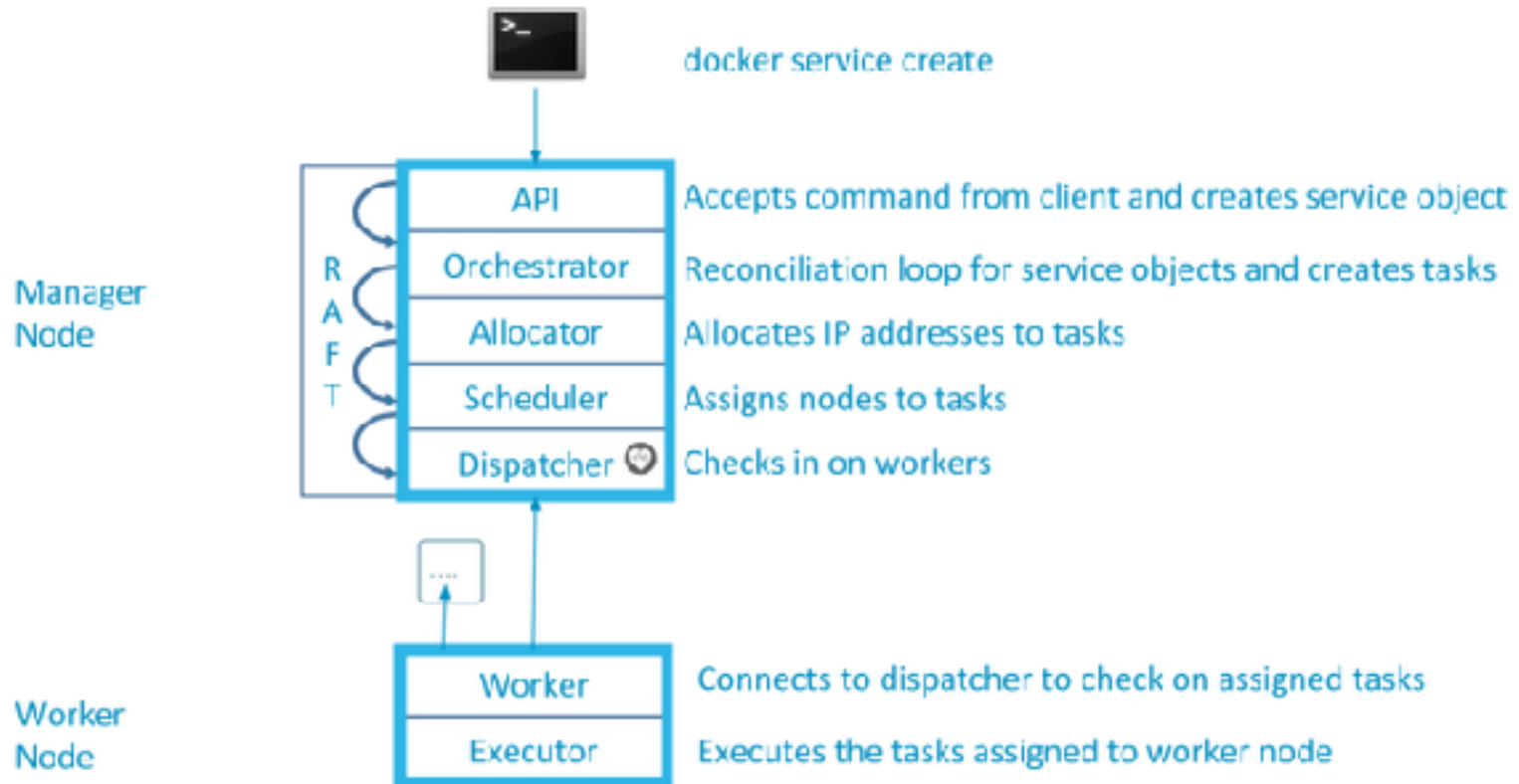
# Docker 1.12 under the hood



## Docker Swarm Communication Internals



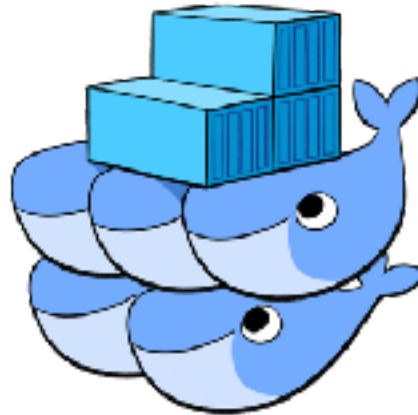
# Docker 1.12 under the hood



# Docker 1.12 under the hood



- Mehr Details zur internen funktionsweise unter <http://collabnix.com/archives/1445>



# Swarmmode Objekte



- **Der Swarm**  
Ein Cluster von Docker Engines
- **Ein Node**  
Die Instanz einer Docker Engine die im Swarm eingebunden ist
- **Services**  
Das Image das innerhalb des Swarms laufen soll
- **Tasks**  
Die Containerinstanzen des Service



# Neue Befehle in der API



- **docker swarm**  
Befehlssatz um einen Swarm zu erstellen
- **docker node**  
Befehle um die Nodes des Swarm-Clusters zu verwalten
- **docker service**  
Befehle um die Images die im Swarm laufen sollen zu verwalten



# Voraussetzungen zum Betrieb



- Um Docker Swarm zu betreiben werden diverse freie Ports benötigt
- Werden benötigt für Kommunikation zwischen ManagerServern und WorkerNodes
- Liste der benötigten Ports und Kommunikationsrichtung unter

<https://docs.docker.com/swarm/plan-for-production/>



# Agenda



- Einleitung
- Docker 1.12 und der Swarm Mode
- Weblogic 12.2.1.2 im Swarm
- Demo
- Q&A



- Warum Weblogic im Swarm betreiben?  
-> z.B. um geclusterte Anwendungen zu testen ohne großen Aufwand, für Entwicklung, oder einfach weil es geht... ;-)
- oracle/docker-images auf github.com  
Oracle hat unter [github.com](https://github.com/oracle/docker-images) eigene Images (bzw. alles zum erstellen dieser) zur Verfügung gestellt, auch aktuellste Version 12.2.1.2
- Viele Beispiele, die als Grundlage verwendet werden können
- aber: Multihost-Beispiele noch ohne die neuen Swarm Funktionen



# Weblogic im Swarm



- Anpassungen notwendig bei den vorhandenen Images
- Etwas höheren Memoryverbrauch berücksichtigen (z.B. bei Adminserver bei Verwaltung vieler ClusterNodes)
- Einige Schritte aus den Beispielen werden nicht mehr benötigt (Consul, Swarm einbinden usw.)



# Agenda

- Einleitung
- Docker 1.12 und der Swarm Mode
- Weblogic 12.2.1.2 im Swarm
- Demo
- Q&A



- 3 Virtuelle Maschinen mit Oracle Linux 7.4 als Nodes
- Docker Version 17.09-ce
- 1 Manager, 2 Worker
- Alle Manager müssen eine IP haben die alle Nodes erreichen können
- Auf dem Manager noch ein docker-repository und einen WebServer zur Darstellung der Sample-App



# Swarm initialisieren



- **Erstellen eines Swarms**

```
docker swarm init --advertise-addr <MANAGER-IP>
```

- **Ermittlung Befehl um einen Worker oder Manager hinzuzufügen**

```
docker swarm join-token worker (oder manager)
```

```
docker swarm join \
```

```
    --token
```

```
SWMTKN-1-4yk2fo3hiuh31922156pa5u34fwhu7cbwglx31dyhzprsa1vsc-  
cjsw29gogjnr5daikwiijh08 \
```

```
    192.168.176.128:2377
```

**Ausgabe dann auf entsprechenden Servern ausführen**



# Overlay-Netz erstellen



- Erstellen des Overlay-Netzes vom Manager

```
docker network create \  
  --driver overlay \  
  --subnet 10.0.9.0/24 \  
  --opt encrypted \  
weblogic-net
```

- Prüfen des Netzes

```
docker network inspect weblogic-net
```

Anzeige IPs der an das Netz angebundenen Container



# Swarm prüfen



- **Nodes anzeigen**

```
docker node ls
```

- **Details zu den Nodes anzeigen**

```
docker node inspect --pretty <nodeid> oder self
```



# Service für Webserver erstellen



- **Erstellen von Service für WebServer**

```
docker service create --name webserver -e  
WEBLOGIC_CLUSTER=wlserver -p 80:80 --constraint  
'node.id == vdm78my7a0uye6kdyoq8uuuf6' --network  
weblogic-net 192.168.176.128:5000/1221-webtier:doag
```

- **Ermittlung Nodeid des Managers**

```
docker node ls
```

- **ManagerNode in Pause-Mode setzen**

```
docker node update --availability pause <nodeid>
```





# Service für WeblogicServer erstellen



- Erstellen von Service für ManagedServer (WLS-ClusterNodes)

```
docker service create --name wlserver --replicas 2  
--network weblogic-net 192.168.176.128:5000/12212-  
appdeploy:doag
```

- Ermittlung Nodeid der Worker

```
docker node ls
```

- tasks auf Worker-Nodes anzeigen

```
docker node ps <nodeid>
```



- Aufruf der Applikation mit  
`http://manager:8001/sample`

## WebLogic on Docker - Request Information

- `getVirtualServerName(): AdminServer`
- `InetAddress.hostname: abf358f2e120`
- `InetAddress.serverAddress: abf358f2e120/10.0.9.5`
- `getLocalAddr(): 10.0.9.5`
- `getLocalName(): abf358f2e120`
- `getLocalPort(): 8001`
- `getServerName(): 192.168.176.128`
- `WLS Server Name: AdminServer`
- `getIpAddOfCurrSrv():`



- Anzahl der Replicas des Service im Swarm erhöhen

```
docker service scale wlserver=4
```

oder

```
docker service update replicas=4 wlserver
```

- Ermittlung Nodeid der Worker

```
docker node ls
```

- tasks auf Worker-Nodes anzeigen

```
docker node ps <nodeid> oder <hostname>
```



# DNS Auflösung innerhalb Swarm



- DNS der Loadbalancing-Adresse entspricht dem Namen des Services, hier `wlserver` (in Container ausführen)

```
docker exec -it -u 0 <CONTAINERID> nslookup wlserver
```

Non-authoritative answer:

Name: `wlserver`

Address: `10.0.9.2`



# DNS Auflösung innerhalb Swarm



- Jeder Node eigenen DNS-Eintrag:

```
docker exec -it -u 0 <containerid> nslookup  
tasks.wlserver
```

Non-authoritative answer:

Name: tasks.wlserver

Address: 10.0.9.3

Name: tasks.wlserver

Address: 10.0.9.8



# DNS Auflösung innerhalb Swarm



- Auf Nodes Overlay-Netz zeigen

```
docker network inspect weblogic-net
— SNIP —
```

```
"Internal": false,
  "Containers": {
    "f17ea11d670e14619366ec4ed6ba1735394350601774980d6954b6b08ec27910": {
      "Name": "wlserver.1.d5mvx4gva36huybpr6owfijhq",
      "EndpointID":
    "43f5461cae44514d7b55a2ea159782cacdc709112577bf37694dc9b62011e653",
      "MacAddress": "02:42:0a:00:09:04",
      "IPv4Address": "10.0.9.4/24",
      "IPv6Address": ""
    }
  }
```



# DNS Auflösung innerhalb Swarm



- Virtuelle IP des internen Loadbalancers anzeigen

```
docker service inspect \
```

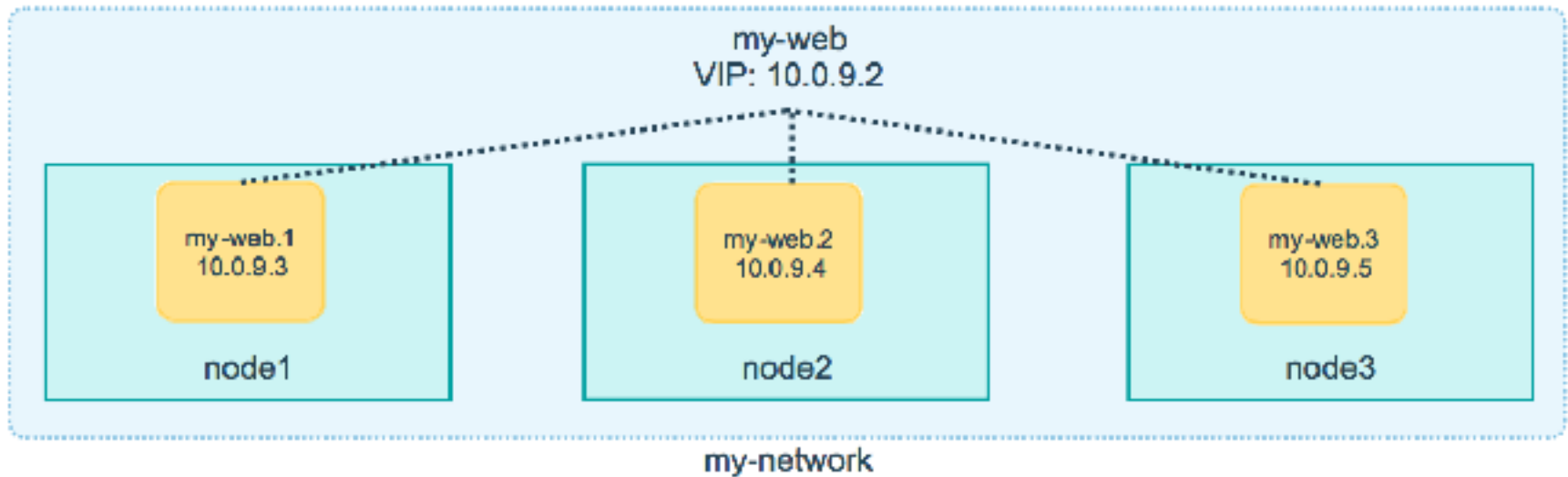
```
--format='{{json .Endpoint.VirtualIPs}}' \
```

```
wlserver
```

```
[{"NetworkID":"2zge1jfjuqiv2w0svst8eqwk7","Addr":"10.0.9.6/24"}]
```



# DNS-Auflösung





- **Für weiteren Ablauf entfernen der Services**

```
docker service rm wlserver
```

- **ManagerServer wieder in Active availability setzen**

```
docker node update availability active  
2ztwqm8p9ilqn7u0baibrbmy
```

- **Service für Adminserver starten (nur auf ManagerNode)**

```
docker service create --name wlsadmin -p 8001:8001 --  
reserve-memory 2G --constraint 'node.id ==  
vdm78my7a0uye6kdyoq8uuuf6' --network weblogic-net  
192.168.176.128:5000/12212-appdeploy:doag
```



- **Service für ManagedServer starten (nur auf WorkerNodes)**

```
docker service create --name wlserver --constraint  
'node.id != vdm78my7a0uye6kdyoq8uuuf6' --replicas 2  
--network weblogic-net 192.168.176.128:5000/12212-  
appdeploy:doag createServer.sh
```

- **Services prüfen**

```
docker service ls  
docker service ps wlserver
```



# Agenda

- Einleitung
- Docker 1.12 und der Swarm Mode
- Weblogic 12.2.1.2 im Swarm
- Demo
- Q&A



# Fragen und Antworten



- Github von Oracle Dockerfiles und Skripte von Bruno Borges  
<https://github.com/oracle/docker-images>
- Dokumentation von Docker  
<https://docs.docker.com/engine/swarm/>
- Blog zu Multihost Networking mit Docker und Weblogic (allerdings Docker Version < 1.12)  
<http://oraclemiddlewareblog.com/2015/05/04/docker-swarm-and-weave-running-a-containerized-weblogic-12c-domain-on-multiple-hosts/>





## Thorsten Wussow

SLIX Gesellschaft für Computersysteme mbH  
Nandlstädter Weg 6  
84072 Au i. D. Hallertau

phone +49 1733208013  
mail [thorsten.wussow@slix.de](mailto:thorsten.wussow@slix.de)  
web [www.slix.de](http://www.slix.de)