

Forms 12c, Reports und WebLogic in Docker betreiben

Holger Lehmann, OPITZ CONSULTING Deutschland GmbH



Das Thema „Virtualisierung“ ist derzeit in aller Mund und die Aufmerksamkeit, die sich darauf richtet, hat besonders in den letzten Jahren deutlich zugenommen.

Virtualisierung bringt durch die bessere Auslastung der Hardware für Unternehmen deutliche Einsparpotenziale mit sich und vereinfacht das Aufsetzen kompletter Entwicklungsumgebungen. Dieser Artikel beschreibt, wie der Autor als Entwickler eine Oracle-Forms-Umgebung auf einem WebLogic-Server in modularen, wiederverwendbaren und transportablen Containern aufbauen und damit eine alte Technologie wie den Forms Stack mit modernen Werkzeugen wie Docker verbinden konnte. Das geht bedeutend schneller, als eine komplette Forms/Reports-Umgebung in einer virtuellen Maschine aufzusetzen. Der Beitrag richtet sich also vor allem an Entwickler, die ebenfalls experimentierfreudig sind und auch selbst eine Oracle-Forms-Entwicklungsumgebung installieren und konfigurieren möchten.

Es geht jetzt darum, die Software-Komponenten Forms 12c, Reports und WebLogic (WLS) in Docker zum Laufen zu bringen. Das auszuprobieren, war schon länger ein großer Wunsch des Autors. Dirk Nachbar [1] hat dafür großartige Vorarbeit geleistet und seine Sources und Anleitungen in

GitHub [2] bereitgestellt. Im Vorfeld haben ihn Robert Cramers [3] und Jan-Peter Timmermann [4] dabei unterstützt.

„Warum ausgerechnet Docker?“, werden sich viele fragen. Die einfache Antwort lautet: „Docker ist ein tolles Tool, weil man nicht für jede neue Instanz einer Entwicklungsumgebung wieder eine neue virtuelle Maschine neu aufsetzen muss.“ Außerdem gewährleisten Container die Trennung und Verwaltung der auf einem Rechner genutzten Ressourcen [5]. Darüber hinaus bieten automatisierte Builds und das Bereitstellen einer Umgebung ohne viele manuelle Aktivitäten einfach viele Vorteile.

Beispiel-Setup

Das gewählte Setup für dieses Vorgehen sieht folgendermaßen aus:

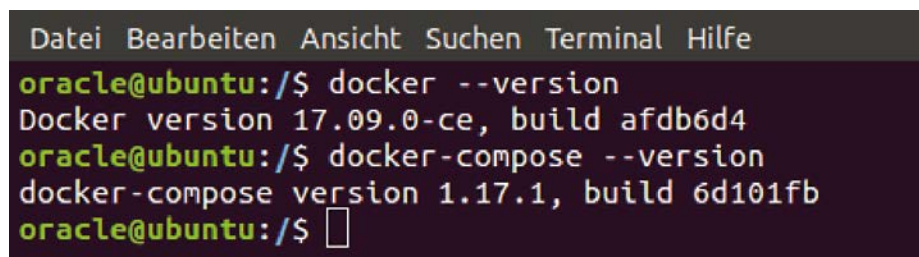
- Ein Laptop mit Windows 7
- Eine virtuelle Maschine mit Ubuntu 17.10 auf dem Laptop
- In Ubuntu laufen Docker und die Container

Die Ubuntu-Maschine von Linux und der WebLogic-Docker-Container teilen sich freigegebene Verzeichnisse, in denen die erstellte WebLogic-Domäne liegt. So kann diese direkt von Ubuntu aus verändert und konfiguriert werden. Ein paar weitere Vorarbeiten sind noch zu leisten, wie die Anlage des Betriebssystem-Users „oracle“ und der Gruppe „oinstall“. Auf dem Ubuntu-System waren noch die Versionen von „docker“ und „docker-compose“ zu aktualisieren, denn die aus dem vorhandenen Software-Repository waren nicht mehr ganz aktuell. Bei ersten Versuchen mit nicht aktuellen Versionen der Tools war das Erstellen der Docker-Container nicht erfolgreich (siehe Abbildung 1).

Setup der Docker-Images

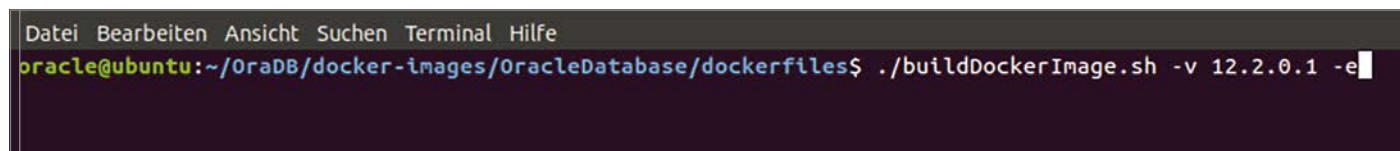
Die Installationsschritte für das Docker-Image entsprechen im Prinzip genau den Schritten einer Installation von Forms und Reports 12c mit einem WebLogic-Server auf einem Host-System. Man braucht dafür Folgendes:

- Eine Datenbank für das Repository (entweder im Docker-Container oder im Host)
- Einen WebLogic-Server in Docker
- Eine Repository-Installation
- Eine Forms/Reports-WebLogic-Domäne



```
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
oracle@ubuntu:/$ docker --version
Docker version 17.09.0-ce, build afdb6d4
oracle@ubuntu:/$ docker-compose --version
docker-compose version 1.17.1, build 6d101fb
oracle@ubuntu:/$
```

Abbildung 1: Kommandos zum Upgrade von „docker“ und „docker-compose“



```
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
oracle@ubuntu:~/OraDB/docker-images/OracleDatabase/dockerfiles$ ./buildDockerImage.sh -v 12.2.0.1 -e
```

Abbildung 2: Kommando zum Bauen der Datenbank als Docker-Image



```
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
oracle@ubuntu:~/Docker/OracleJava/java-8$ ./buildDockerImage.sh
```

Abbildung 3: Kommando zum Bauen des JDK 8u151


```
oracle@ubuntu:~/Docker/OracleJava/java-8$ docker images
```

| REPOSITORY | TAG | IMAGE ID | CREATED | SIZE |
|-------------------------------|-------------|--------------|--------------|--------|
| oracle/database | 12.2.0.1-ee | af862e9c3077 | 2 days ago | 13.3GB |
| localhost/oracle/formsreports | 12.2.1.3 | 480fada974f0 | 2 days ago | 14.2GB |
| oracle/fmw-infrastructure | 12.2.1.3 | 86b98d025633 | 2 days ago | 6.17GB |
| oracle/serverjdk | 8 | cf6c22158ffc | 2 days ago | 614MB |
| oraclelinux | latest | f426f15a5793 | 9 days ago | 229MB |
| oraclelinux | 7-slim | 9870bebf1d5 | 9 days ago | 118MB |
| sath89/oracle-12c | latest | f52b86b93aab | 2 months ago | 5.7GB |
| sath89/oracle-xe-11g | latest | 04851454491b | 4 months ago | 792MB |

```
oracle@ubuntu:~/Docker/OracleJava/java-8$
```

Abbildung 4: Kommando zum Auflisten aller Docker-Images

```
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
oracle@ubuntu:~/Docker/OracleFMWInfrastructure/dockerfiles$ ./buildDockerImage.sh -v 12.2.1.3
```

Abbildung 5: Kommando zum Bauen des WebLogic-Server-Image

```
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
oracle@ubuntu:~/Docker/OracleFormsReports/dockerfiles$ ./buildDockerImage.sh -v 12.2.1.3
```

Abbildung 6: Kommando zum Bauen des Forms/Reports-Image

Für das Installationsbeispiel wurden diese Komponenten in mehreren Docker-Images installiert und dabei der Anleitung von Dirk Nachbar aus seinem GitHub-Repository [2] gefolgt.

Aufbau des Datenbank-Image

Bei der Installation der Datenbank dient eine Oracle Standard oder Enterprise Edition als Repository-Datenbank für die WebLogic-Domäne. Dafür wurde ein Git Clone des offiziellen Oracle-Docker-GitHub [6] erstellt und mit „./buildDockerImage.sh -v 12.2.0.1 -e“ ein Image der Enterprise Edition gebaut (siehe Abbildung 2). Nach dem erfolgreichen Erstellen kann der entsprechende Docker-Container zum Beispiel über „docker run --name oracle12-ee -p 1521:1521 -p 5500:5500 -p 8080:8080 -e ORACLE_PWD=neuesPasswort -v /opt/oracle/oradata:/u01/app/oracle/oradata oracle/database:12.2.0.1-ee“ gestartet werden.

```
# NodeManager
DC_NM_LISTENADDRESS=`hostname -f`
DC_NM_TYPE=SSL
DC_NM_PORT=5556
DC_NM_USERNAME=nodemanager
DC_NM_PWD=welcome1

# Repository Connect
DC_DBUSER=sys
DC_DBPWD=oracle
DC_DBROLE=SYSDBA
DC_COMPONENTPWD=oracle
DC_SCHEMA_PREFIX=FRTEST
# DB Host, IP des Containers
DC_DB_HOST=172.17.0.2
DC_DB_PORT=1521
DC_DB_SERVICE=XE
DC_DB_OMF=false
DC_DB_USER_PW=oracle
DC_PWDFILE=/tmp/passwords.txt
```

Abbildung 7: Auszug aus der Konfigurationsdatei „setenv.sh“

```
docker0: flags=4163<UP,BR
    inet 172.17.0.1
```

Abbildung 8: Ansicht der IP-Adresse der Komponente „docker0“

Aufbau des WebLogic-Image

Der Aufbau des WebLogic-Image beginnt mit dem „Git Clone“ von Dirks GitHub-Repository [2] und erfolgt in mehreren

Schritten. Zuerst ging es um den Build des „OracleLinux:latest“ mit dem Oracle Java Development Kit (JDK) 8u151. Dafür wurde das JDK 8u151 tar.gz in den Ordner

„OracleJava/java-8“ gelegt und ein „./buildDockerImage.sh“ angefertigt (siehe Abbildung 3). Das resultierende Endprodukt ist ein Docker-Image mit dem Tag „oracle/

```
... <BEA-000365> <Server state changed to RUNNING.># ... <BEA-000365> <Server state changed to RUNNING.>
```

Abbildung 9: Ansicht der Meldung nach dem Start der WebLogic-Domäne

serverjdk“. Anschauen kann man sich die vorhandenen Images mit dem Befehl „docker images“ (siehe Abbildung 4).

Im nächsten Schritt wird ins Verzeichnis „OracleFMWInfrastructure/dockerfiles“ gewechselt und ein WebLogic-Image erstellt. Dafür ist die WebLogic-Installati-

onsdatei „fmw_12.2.1.3.0_infrastructure_Disk1_1of1.zip“ erforderlich, die von Oracle heruntergeladen und im Unterordner „OracleFMWInfrastructure/dockerfiles/12.2.1.3“ abgelegt wurde. Mit „./build-DockerImage.sh -v 12.2.1.3“ wurde der Build gestartet. Das Resultat war ein Oracle

WebLogic Server Infrastructure 12.2.1.3.0 (siehe Abbildung 5).

Im letzten Schritt wurde das WebLogic-Server-Infrastructure-Image um die Forms- und Reports-Sourcen erweitert. Dafür wechselt man ins Verzeichnis „OracleFormsReports/dockerfiles“ und legt dort die Installations-Dateien „fmw_12.2.1.3.0_fr_linux64.bin“ und „fmw_12.2.1.3.0_fr_linux64-2.zip“ im Unterordner „OracleFormsReports/dockerfiles/12.2.1.3“ ab. Der Befehl „./build-DockerImage.sh -v 12.2.1.3“ startet den Image-Erstellungsprozess (siehe Abbildung 6). Am Ende steht ein Image mit dem Namen „localhost/oracle/formsreports TAG: 12.2.1.3“ zur Verfügung.

Konfigurieren der WebLogic-Domäne

Die WebLogic-Domäne wird im „Silent Mode“ erstellt, auf Benutzer-Eingaben und grafische Oberfläche wird dabei also komplett verzichtet. Dirk Nachbar hat im Verzeichnis „OracleFormsReports/samples“ [2] eine Konfigurationsdatei „setenv.sh“ vorbereitet, die die nötigen Umgebungsvariablen für die Domänen-Konfigurationsparameter setzt (Abbildung 7 zeigt einen Auszug).

Diese Datei ist entsprechend anzupassen; sehr wichtig sind die Eintragungen im Abschnitt „#Repository Connect“ für den Connect des Repository Configuration Utility (RCU) gegen diese Datenbank. Hier tappte der Autor zuerst in eine Falle und gab für „DC_DB_HOST“ die falsche IP an. Damit funktionierte die Erstellung der Domäne zunächst nicht. Dann stieß er auf „ifconfig“ auf dem Docker-Host. Dort gibt es ein Netzwerk namens „docker0“ und unter der „inet“ findet sich die richtige IP für den Datenbank-Container (siehe Abbildung 8).

Im Anschluss konnte die Datenbank auf dem ersten Container gestartet und die WebLogic-Domäne sowie der Admin-Server erzeugt werden. Die Variablen aus „setenv.sh“ werden dafür gespeichert, exportiert und vom Programm „docker-compose“ genutzt: „source ./setenv.sh docker-compose up -d frfmw; docker logs

Abbildung 10: Login-Bildschirm des Enterprise Manager

```
<Dec 8, 2017 4:48:54,783 PM UTC> <Warning> <NodeManager> <BEA-300057>
<Starting server MS_FORMS on machine AdminServerMachine at ubuntu:5556>
```

Abbildung 11: Console des Enterprise Manager beim Start für „MS_FORMS“

Abbildung 12: Start für MS_FORMS aus dem Enterprise Manager erfolgreich

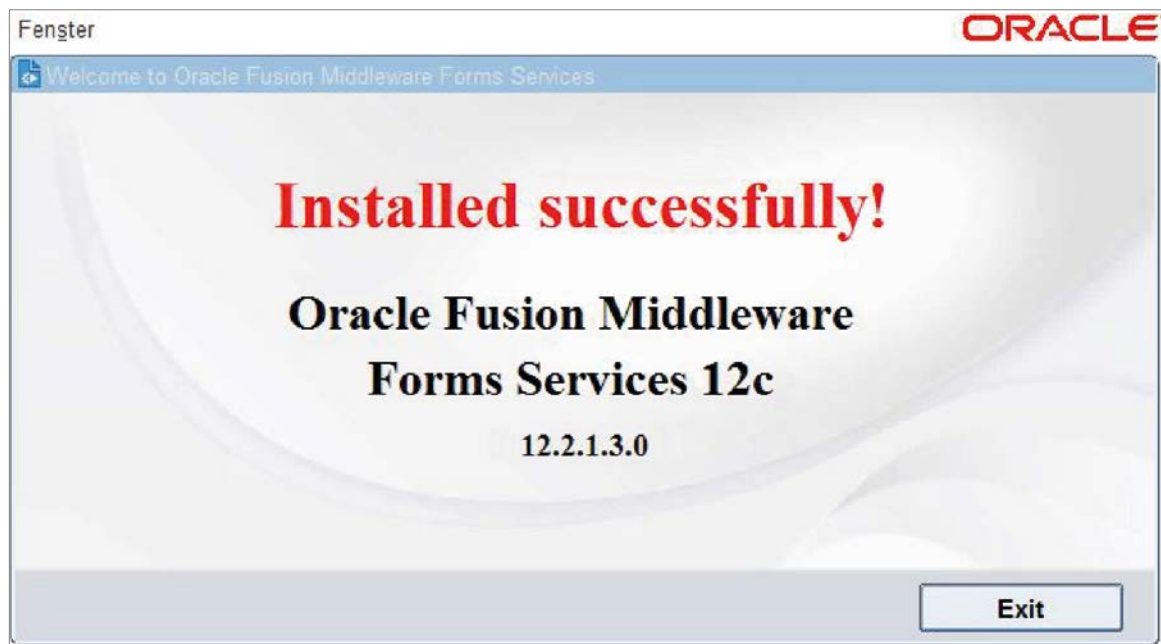


Abbildung 13: Aufruf der Testseite für Forms erfolgreich

```
docker exec -ti frfmw /bin/bash
cd /opt/oracle/user_projects/domains/<DOMAIN_NAME>/bin
./startComponent.sh <REPORTS_SERVER_NAME>
exit
```

Abbildung 14: Kommando zum Start eines Reports-Servers

ren sollte. Nach Ansicht des Autors sollte der nächste Schritt sein zu prüfen, ob die hierbei erstellten Docker-Container mit dem Oracle Container Cloud Service auch in der Oracle-Cloud laufen können.

frfmw -f". Wird am Ende des Startens der WebLogic-Domäne „<BEA-000365> <Server state changed to RUNNING.>" angezeigt, ist alles gut verlaufen (siehe Abbildung 9).

Nun kann der Oracle Enterprise Manager über die URL „http://localhost:7001/em“ im Browser aufgerufen werden (siehe Abbildung 10). Im Enterprise Manager lassen sich nun die Managed Server „MS_FORMS“ und „MS_REPORTS“ für Forms und Reports starten (siehe Abbildung 11 und 12).

Wenn man nun noch die Ports von der Ubuntu VM entsprechend weiterleitet, kann man im Browser auch die Forms-Testseite aufrufen: „http://localhost:9001/forms/frmservlet“ (siehe Abbildung 13). Das ist noch nicht alles: Man könnte zum Beispiel auch einen zuvor erstellten Reports-Server starten (siehe Abbildung 14).

Da alle Konfigurationsdateien der WebLogic-Domäne in einem Verzeichnis auf dem Docker-Host zugänglich sind, kann man sie dort auch bequem bearbeiten. Alle Änderungen sind persistiert (wie „default.env“, „formsweb.cfg“, „rwserver.conf“, „httpd.conf“ etc.). Auch die neu-

en Oracle Forms Application Deployment Services mit Forms 12.2.1.3.0 lassen sich nutzen. Dafür sei auf einen Blick in das „readme“-File von Dirk Nachbar in GitHub verwiesen [7].

Fazit

Der Autor ist erstmal sehr begeistert von diesem Vorgehen. Nach einem Neustart des Ubuntu-Betriebssystems kann er nun mit zwei einzelnen Docker-Befehlen die Datenbank und den WLS-Admin-Server starten und eine zuvor erstellte Umgebung wiederherstellen. Hier ist es auch von Vorteil, dass die Forms- und WebLogic-Spezialisten Dirk Nachbar [1], Robert Crammes [3] und Jan-Peter Timmermann [4] die ganze Domänen-Erstellung in Skriptform realisiert haben, damit man dafür ohne GUI auskommt.

Doch Achtung: Die Umgebung, um die es hier geht, wird derzeit nicht von Oracle unterstützt und ist daher auch nicht für produktive Umgebungen gedacht. Was nicht heißt, dass man dieses Vorgehen nicht trotzdem nutzen und perfektionie-

Quellen

- [1] Dirk Nachbar Blog: <http://dirknachbar.blogspot.de>
- [2] Dirk Nachbar GitHub: <https://github.com/DirkNachbar/Docker>
- [3] Robert Crammes Blog: <http://robertcrammes.blogspot.ch>
- [4] Jan-Peter Timmermann Blog: <https://jan-peter.me>
- [5] Docker bei Wikipedia: [https://de.wikipedia.org/wiki/Docker_\(Software\)](https://de.wikipedia.org/wiki/Docker_(Software))
- [6] Oracle Docker GitHub: <https://github.com/oracle/docker-images>
- [7] Dirk Nachbar GitHub FADS readme: <https://github.com/DirkNachbar/Docker/blob/master/README.md>



Holger Lehmann
holger.lehmann@opitz-consulting.com