

Ex 03: Analyse Swisscom & Cloud Application Design

Cloud Solutions

**Abteilung Informatik
Hochschule für Technik Rapperswil**

Autoren: Andreas Stalder, David Meister
Datum: 15. Mai 2017

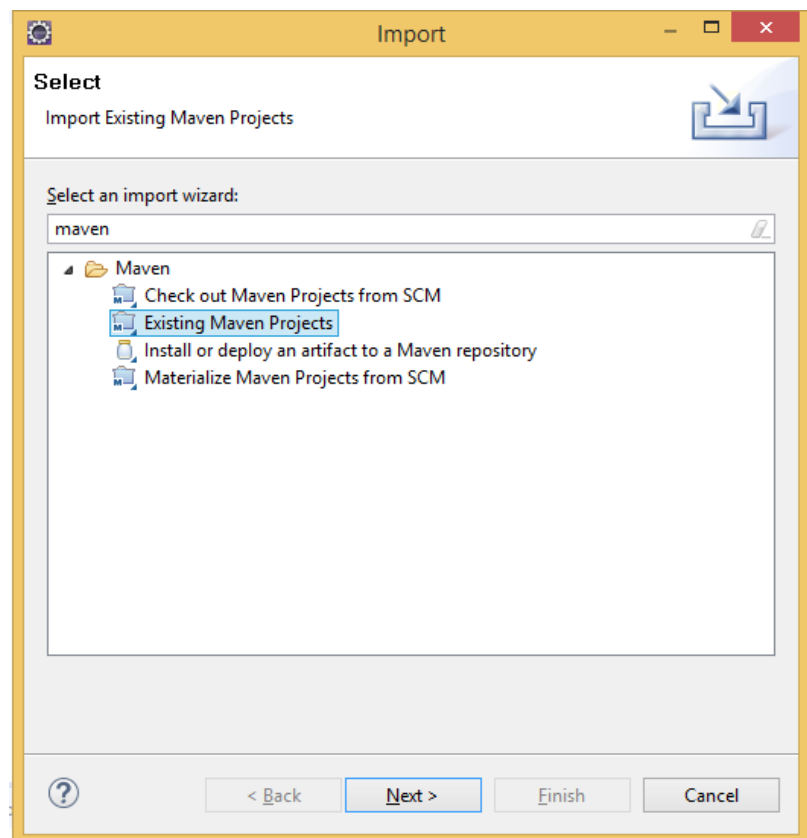
Inhaltsverzeichnis

1 Hands-On: DDD Sample Application	3
1.1 Import der DDD Applikation in die IDE	3
1.2 Deployment in die Swisscom PaaS Cloud	6
1.3 Fazit	11
2 Hands-On: Erweiterung um relationale Datenbank	12
Abbildungsverzeichnis	16

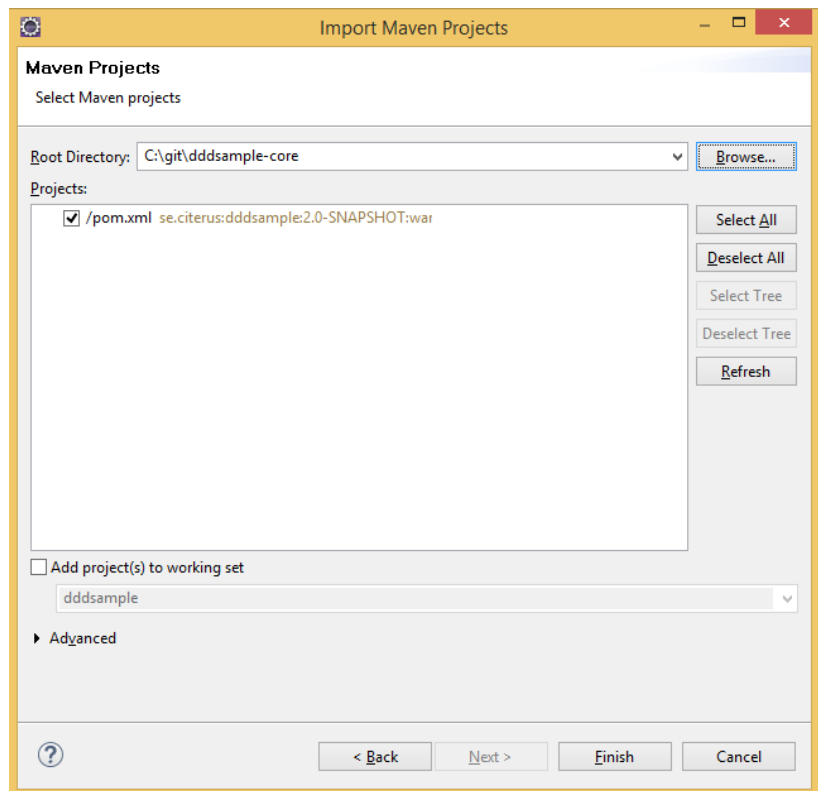
1 Hands-On: DDD Sample Application

1.1 Import der DDD Applikation in die IDE

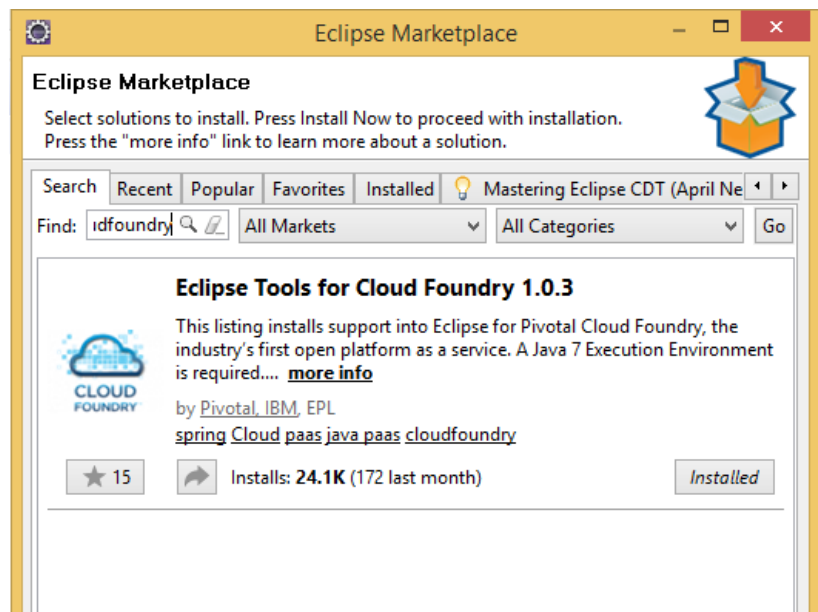
Die Vorlage Applikation kann in Eclipse als bestehendes Maven Projekt importiert werden.



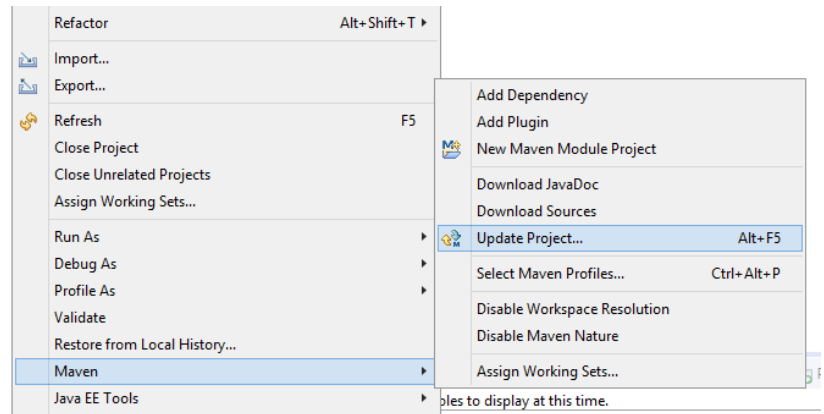
Im Projektverzeichnis die pom.xml Datei auswählen.



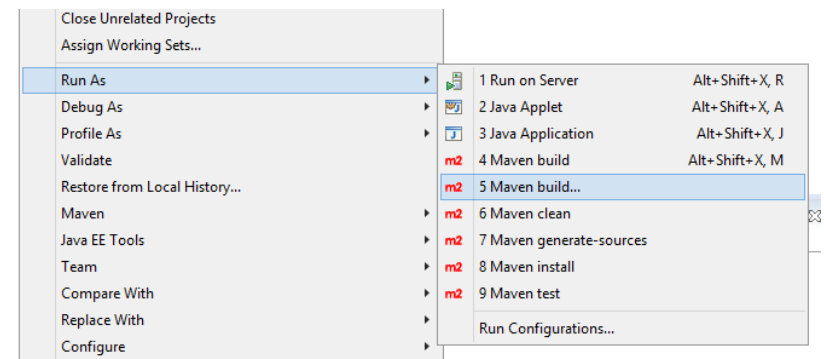
Die Swisscom PaaS Cloud arbeitet mit Cloud Foundry. Das Cloud Foundry Plugin für Eclipse kann über den Marketplace installiert werden.



Nun müssen die Maven Dependencies geladen werden.

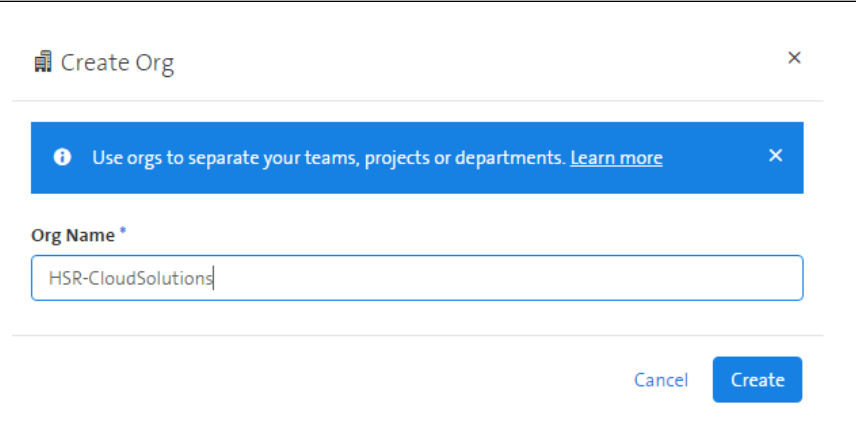
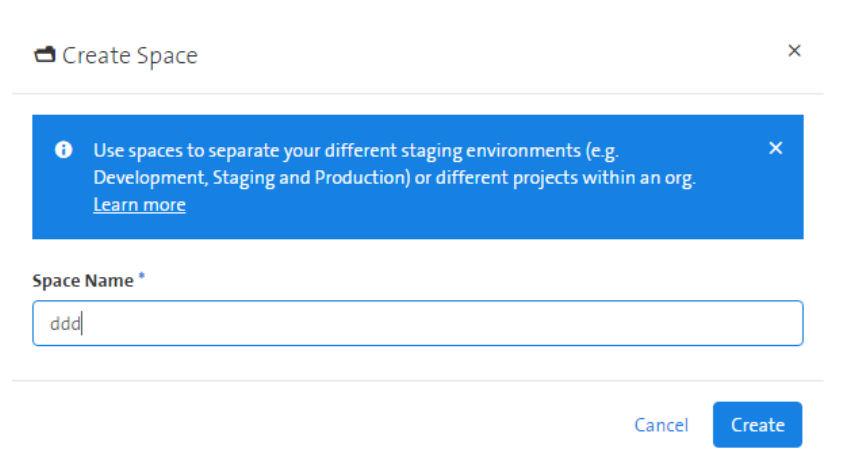


Jetzt kann der Build-Prozess gestartet werden.

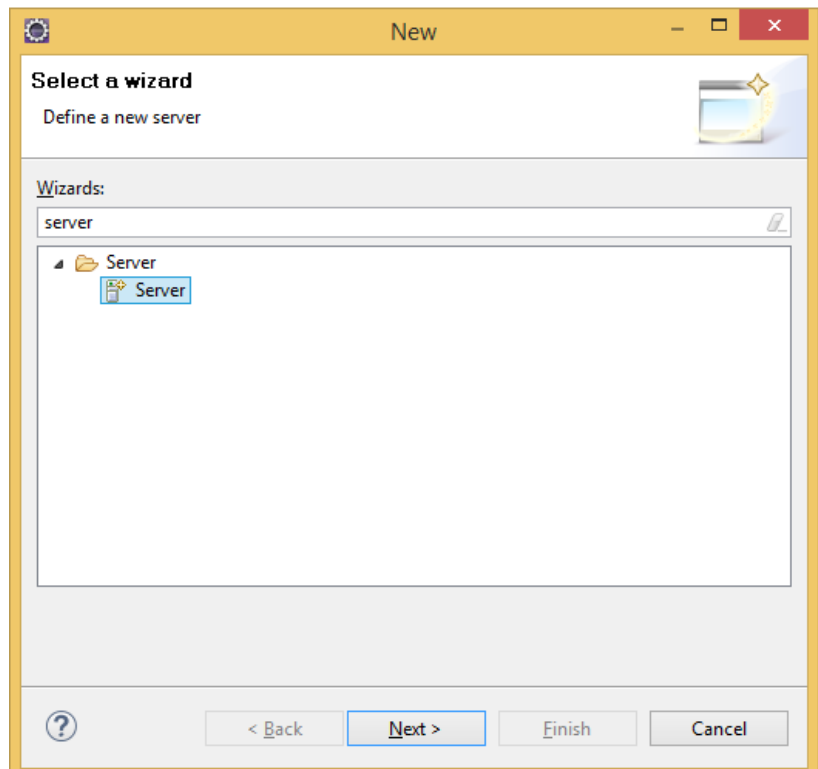


1.2 Deployment in die Swisscom PaaS Cloud

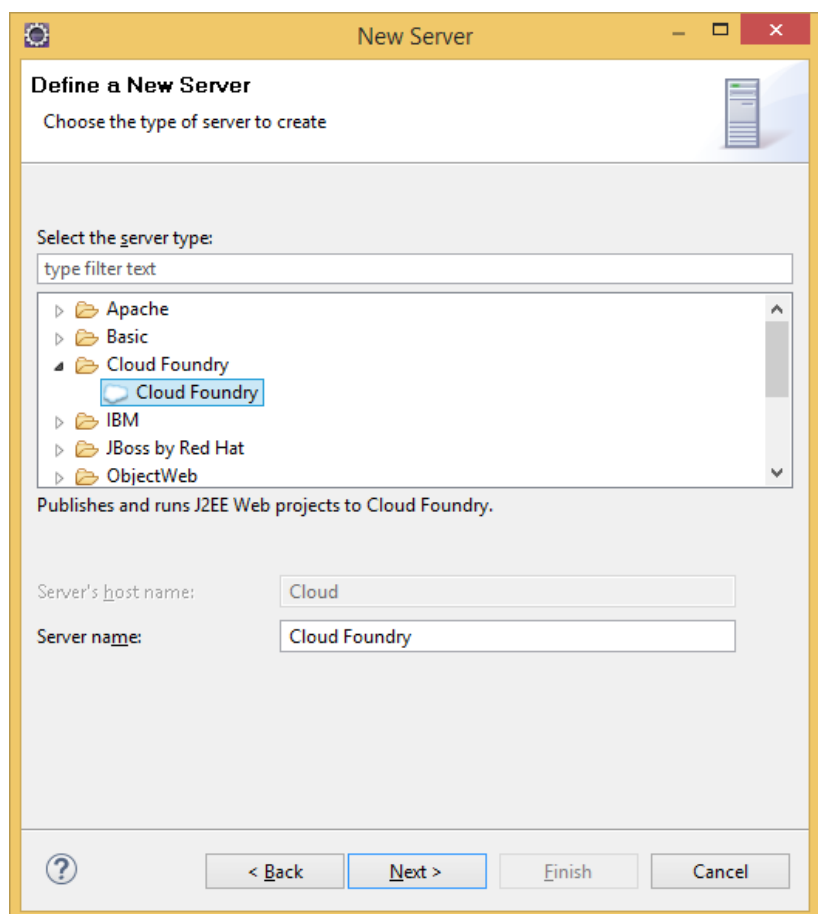
Bevor die Applikation in die Cloud gepusht werden kann muss ein Space angelegt werden. Danach kann die ddd-App in diesen Space deployed werden.

<p>Eine Organisation kann mehrere Spaces enthalten. Eine Applikation wird in einen Space deployed. Wir erstellen deshalb eine neue Org.</p>	
<p>Als Space Name wählen wir ddd (Name der App).</p>	

In Eclipse muss ein neuer Server erfasst werden. Anstatt Tomcat verwenden wir nun einen Cloud Foundry Server



Durch das zuvor im Marketplace installierte Plugin kann nun ein Cloud Foundry Server ausgewählt werden.



Die Logindaten des Swisscom Application Cloud Accounts müssen hier hinterlegt werden. Die URL „https://api.lyra-836.appcloud.swisscom.com“ muss erfasst werden.

The screenshot shows a 'New Server' window with a yellow title bar. Inside, the 'Cloud Foundry Account' section is active. It contains a checkbox for 'Use a one-time password to login (SSO)', an 'Email' field with 'dmeister@hsr.ch', a 'Password' field with masked characters, and a 'URL' dropdown menu showing 'Swisscom - https://api.lyra-836.appcloud.swisscom.com'. Below these fields are buttons for 'Validate Account', 'Register Account...', and 'Sign Up'. At the bottom of the window are navigation buttons: '?', '< Back', 'Next >', 'Finish' (highlighted in blue), and 'Cancel'.

New Server

Cloud Foundry Account
Press 'Validate Account', 'Next', 'Finish' to validate credentials.

Account Information

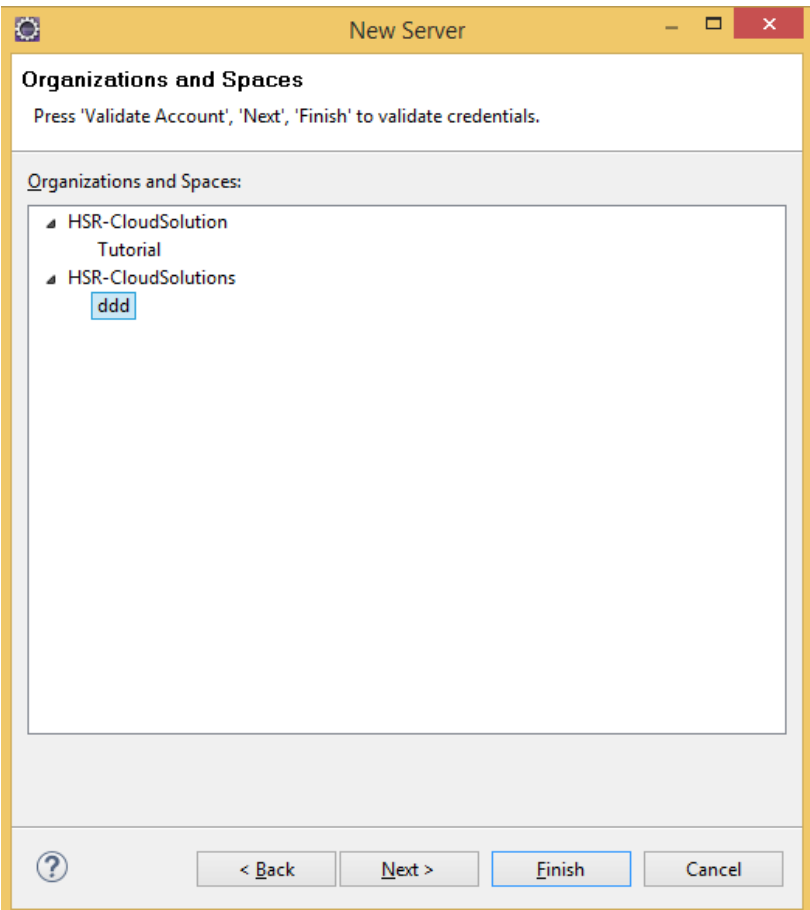
☐ Use a one-time password to login (SSO)

Email:

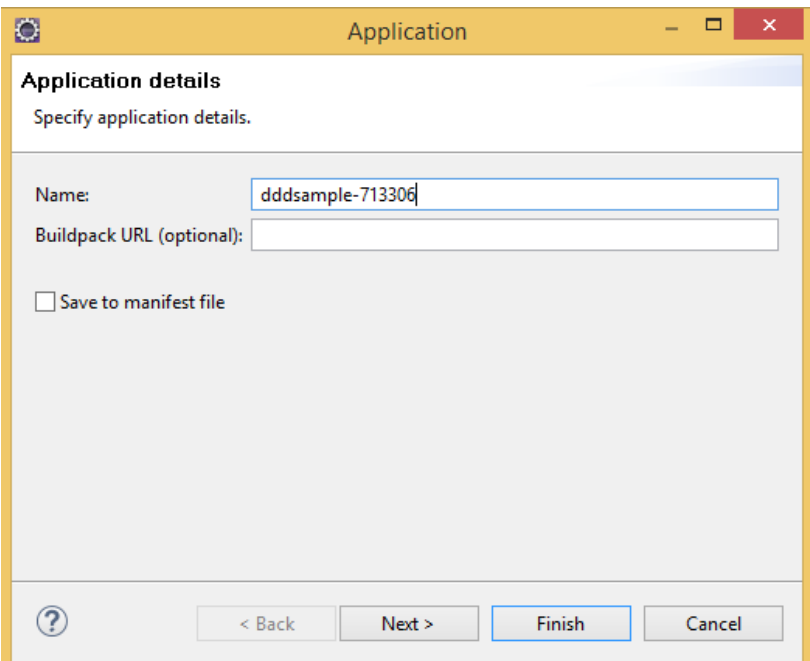
Password:

URL:

Die zuvor erstellte Org und Space sollten nun erscheinen. Den gewünschten Space auswählen um fortzufahren.



Den App Namen kann man belassen oder falls gewünscht anpassen.



Falls benötigt kann das Memory Limit erhöht werden.

Application

Launch deployment

Specify the deployment details

Subdomain:

Domain:

Deployed URL:

Memory Limit (MB):

☒ Start application on deployment

Nun startet der Deployment Prozess, nach kurzer Zeit ist er erfolgreich abgeschlossen.

Publishing to Cloud Foundry...

Publishing to Cloud Foundry...

Getting stats - dddsample-713306

☐ Always run in background


Markers Properties Servers Data Source Explorer Snippets Console

Cloud Foundry--HSR-CloudSolutions--ddd--dddsample-713306

```
Checking application - dddsample-713306
Generating application archive - dddsample-713306
Pushing application - dddsample-713306
Creating application - dddsample-713306
Application successfully pushed
Starting application - dddsample-713306
[Application Running Check] - Checking if application is running - dddsample-713306. Please wait...
```

Auf der Application Cloud Console erscheint nun die laufende App.

HSR-CloudSolu... / d... / dddsample-71... Estimated Monthly Cost: CHF 12.00

 dddsample-713306 Started

Instances - 1 + Memory Limit - 512 MB +

Disk Limit 1 GB

Restart Restage Stop Delete

Instances Events Logs Env Upload Routes

Services

#	STATUS	SINCE	CPU	MEMORY	DISK
0	Running	a minute	0.3%	358.4 MB	160.2 MB

1.3 Fazit

Der grundlegende Teil dieser Aufgabe war mit der Swisscom Application Cloud sehr angenehm zu lösen. Durch die Online-Dokumentation wird man gut durch das Deployment durchgeführt. Die Applikation selbst, respektive der Code musste nicht angepasst werden.


Anfangs hatten wir der Ausführung von gewissen Funktionen. Es stellte sich schnell heraus, dass wir vergessen haben, die Maven Dependencies zu laden und das Projekt zu builden.

2 Hands-On: Erweiterung um relationale Datenbank

Die DDD Sample App arbeitet standardmässig mit HyperSQL DB (HSQLDB). Dies ist in der Datei `jdbc.properties` unter `/src/main/resources` ersichtlich.

Die Swisscom Application Cloud bietet keinen MySQL Service. Stattdessen wird MariaDB als Vertreter relationaler Datenbank angeboten.

MariaDB muss auf dem zuvor erstellten Space (ddd) hinzugefügt werden. Für diese Aufgabe reicht die kleinste Variante mit 1GB Storage und 10 Connections.

 Create MariaDB ×

Service Name *


Service Plan *

1GB Storage Capacity 10 Connections High Availability Small	8GB Storage Capacity 15 Connections High Availability Medium	16GB Storage Capacity 100 Connections High Availability Large
---	---	--

Estimated Monthly Cost **CHF 6.00**

Sobald erstellt, müssen im Reiter „Service Keys“ noch Zugriffsschlüssel erstellt werden. Diese müssen dann in der verwendeten App zur Authentifizierung hinterlegt werden.

HSR-CloudSolutions / ddd / RelationalDB Estimated Monthly Cost: CHF 6.00

 RelationalDB Started

[How to manage](#) [Delete](#)


General Events Backups **Service Keys** Apps

SERVICE KEYS ? [+ Create Service Key](#)

No service keys yet...

Den Keys kann ein beliebiger Name hinterlegt werden.

HSR-CloudSolutions / ddd / RelationalDB Estimated Monthly Cost: CHF 6.00

 RelationalDB Started

[How to manage](#) [Delete](#)

General Events Backups **Service Keys** Apps

SERVICE KEYS ?

[X Cancel](#) [✓ Save](#)

No service keys yet...

Die benötigten Informationen liegen nun im JSON Format da. Die private IPv4 Adresse im host-Feld verrät, dass ein externer Zugriff ausserhalb der Swisscom Application Cloud nicht möglich ist. Sowohl Username und Passwort sind ein zufällig generierter String.

ddd-keys

```
{
  "host": "10.0.20.18",
  "hostname": "10.0.20.18",
  "port": 3306,
  "name": "CF_2035A39F_EB1D_4CB7_BAB7_21DC96F998BA",
  "database": "CF_2035A39F_EB1D_4CB7_BAB7_21DC96F998BA",
  "username": "7XmyQy8z2SimelYN",
  "password": "fz02XBkTccGyhCAD",
  "database_uri": "mysql://7XmyQy8z2SimelYN:fz02XBkTccGyhCAD@10.0.20.18:3306/CF_2035A39F_EB1D_4CB7_BAB7_21DC96F998BA?reconnect=true",
  "uri": "mysql://7XmyQy8z2SimelYN:fz02XBkTccGyhCAD@10.0.20.18:3306/CF_2035A39F_EB1D_4CB7_BAB7_21DC96F998BA?reconnect=true",
  "jdbcUrl": "jdbc:mysql://10.0.20.18:3306/CF_2035A39F_EB1D_4CB7_BAB7_21DC96F998BA?user=7XmyQy8z2SimelYN&password=fz02XBkTccGyhCAD"
}
```

Der bestehenden App kann nun der zuvor erstellte DB Service gebunden werden.

HSR-CloudSolu... / d... / dddsample-71... Estimated Monthly Cost: CHF 12.00



dddsample-713306

Started

Instances − 1 +

Memory Limit − 512 MB +

Disk Limit 1 GB

Restart

Restage

Stop

Delete

Instances

Events

Logs

Env

Upload

Routes

Services

BOUND SERVICES ?

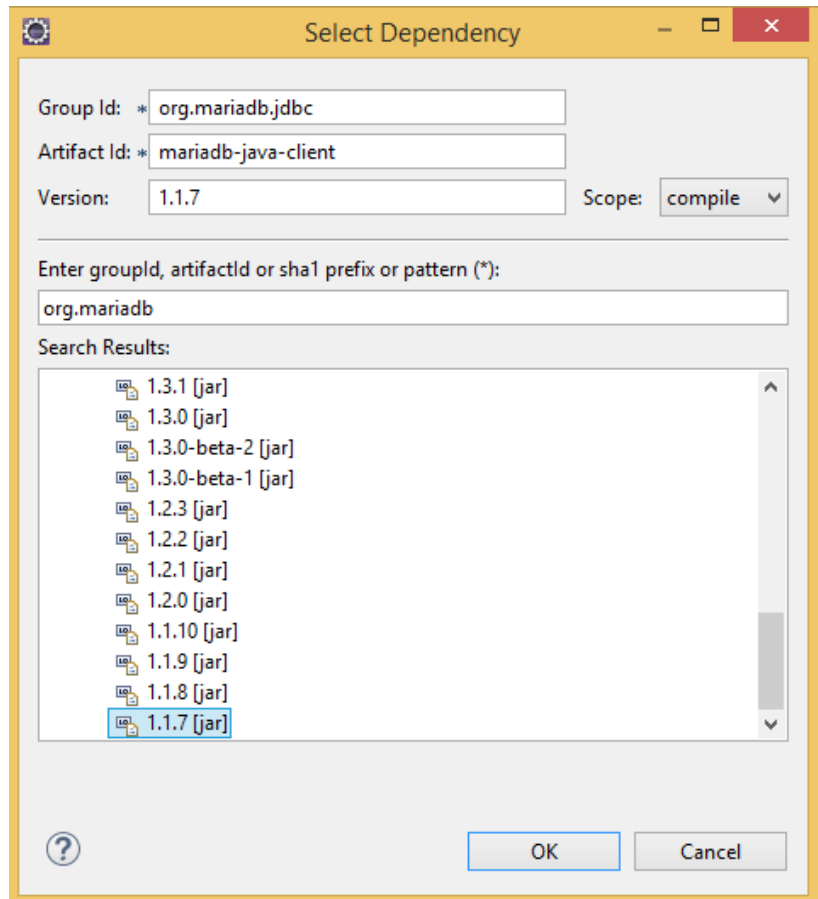
+ Bind Service



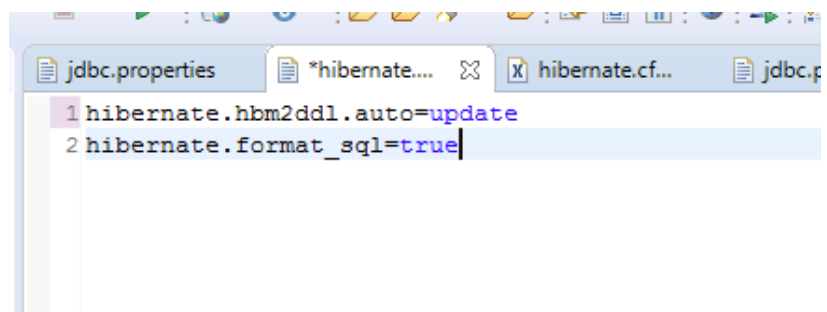
RelationalID.1 bound app

Started

Wir benötigen für die App nun die MariaDB Dependencies, welche wir einfach mit Maven hinzufügen können. Wir verwenden die als funktionierend empfohlene Version 1.1.7.



Damit die Änderungen in der Datenbank auch nach dem Neustart persistent sind, müssen wir eine Anpassung der Konfiguration im File hibernate.properties vornehmen. Im Feld hibernate.hbm2ddl.auto müssen wir die Einstellung von „create drop“(nicht persistent) nach „update“ändern.



Die Änderungen müssen nun mit Maven Update, resp. Maven Build angepasst und neu gepusht werden.

Abbildungsverzeichnis