

# Gruppenarbeit 1 - Cloud Fundamentals beim Provider

Alexander van Schie & Oli Dias

March 11, 2019

# Contents

<b>1 Hands-On: Hello (Cloud) World</b>	<b>3</b>
1.1 Installationsanleitung . . . . .	3
1.1.1 Openshift Account erstellen . . . . .	3
1.1.2 Erstellen, Builden und Deployment der Applikation mit dem Web-UI . . . . .	5
1.1.3 Vorarbeit - Installation der Openshift CLI . . . . .	6
<b>2 Analyse: OSSM-Definition</b>	<b>7</b>
2.1 On-demand & Self-service . . . . .	7
2.2 Scalable . . . . .	8
2.3 Measurable . . . . .	9
<b>3 Konzept: Cloud Computing Patterns</b>	<b>9</b>
<b>4 Hands-On: Self Information</b>	<b>10</b>
<b>5 Analyse: Preisrecherche</b>	<b>10</b>
<b>6 Analyse: Preisvergleich eigenes Hosting, IaaS und PaaS</b>	<b>10</b>

# 1 Hands-On: Hello (Cloud) World

## 1.1 Installationsanleitung

Das Ziel ist es, eine Asp.NET-Core Hello-World Applikation mittels Openshift Online zu builden und deployen. Am Schluss dieser Installationsanleitung sollte dies möglich sein.

Für das Einrichten von Openshift Online müssen grob folgende Schritte durchgeführt werden:

- Account und Projekt auf der Plattform erstellen
- GitHub Repository der Asp.NET-Core Applikation mit Openshift Projekt verbinden
- Applikation auf Openshift builden
- Applikation auf Openshift deployen

### 1.1.1 Openshift Account erstellen

Zuerst muss ein Account auf <https://manage.openshift.com/signin> erstellt werden. Danach kann zwischen den in Abbildung 1 vorgeschlagenen Plänen ausgewählt werden. Wir benutzen den Openshift Online Pro Plan (30-day trial).

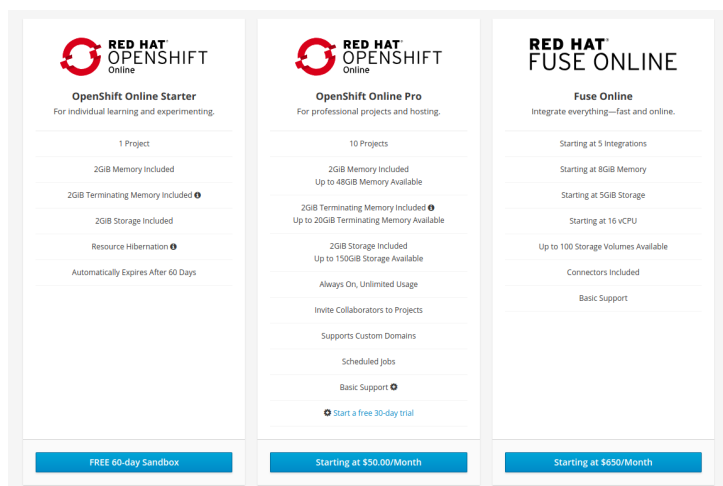


Figure 1: Gewählter Plan Openshift

Dass der Account verifiziert werden kann, muss eine Telefonnummer hinterlegt werden, auf welche darauffolgend einen Bestätigungscode zugeschickt wird.

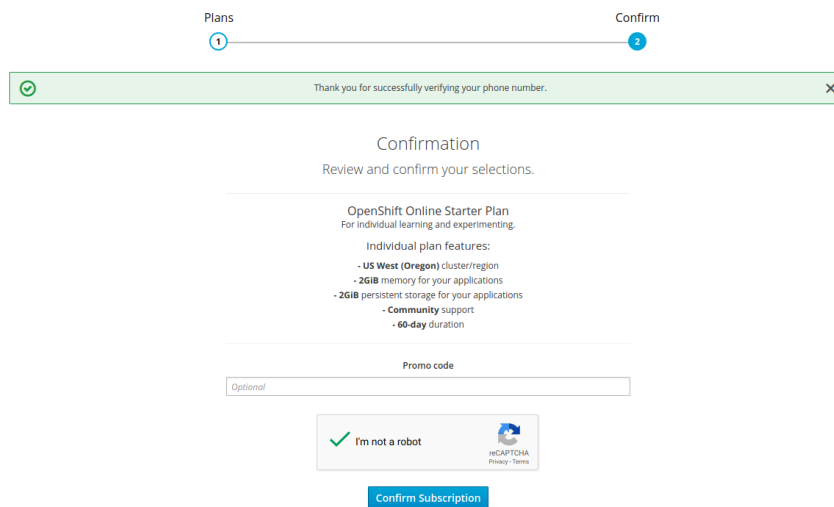


Figure 2: Übersicht des abgeschlossenen Plans

Wurde diese eingegeben und verifiziert, erscheint eine Übersicht über das bestellte Produkt wie in Abbildung 2 angezeigt. Daraufhin kann die Subscription bestätigt werden.

Kurz nach dem Bestätigen sollte ein Bestätigungsmail eintreffen. Darauffolgend kann bereits die Web Console geöffnet werden. Es wird ein Katalog mit allen Produkten von OpenShift Online dargestellt (Abbildung 3).

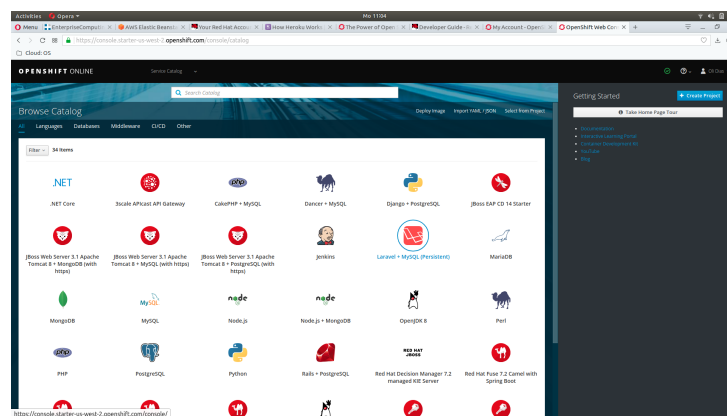


Figure 3: Katalog von OpenShift

Das Erstellen des OpenShift Accounts ist somit abgeschlossen und die Plattform ist für das Builden und Deployen von Applikationen bereit.

### 1.1.2 Erstellen, Builden und Deployment der Applikation mit dem Web-UI

In der Web-Console können wir nun auf .NET Core Projekt klicken. Daraufhin erscheint ein Wizard, dem wir Schritt für Schritt folgen können. Falls das GitHub-Repository schon während dem Wizard hinzugefügt werden soll, muss es bereits existieren und sichtbar sein.

Entsprechende Felder müssen gemäss Abbildung 4 ausgefüllt sein. Vorsicht ist bei der .NET Version geboten; wir verwenden die Version 2.2 von .NET Core.

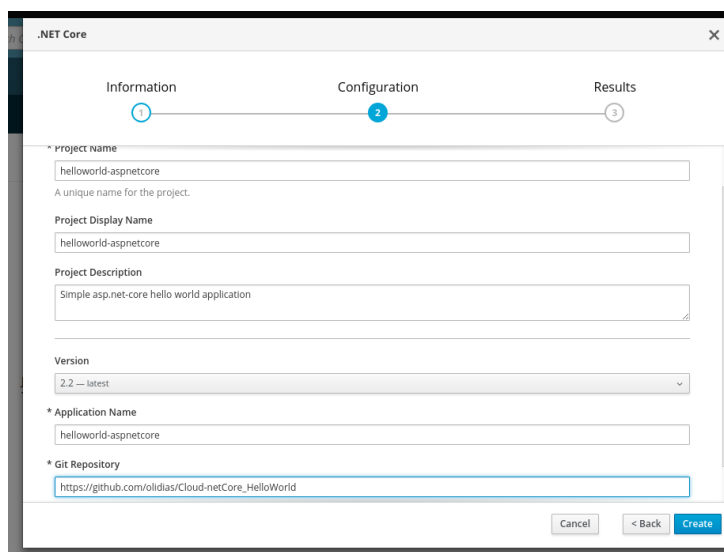


Figure 4: Konfiguration des neuen Projektes

Sobald das Projekt in Openshift erstellt wurde, startet der Build. Womöglich

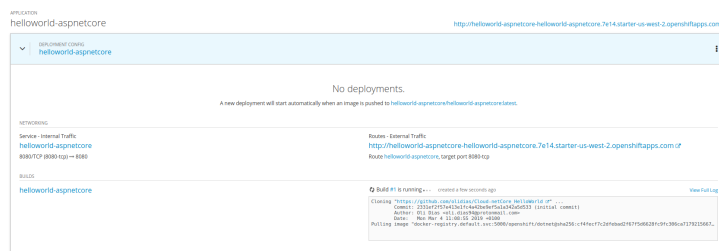


Figure 5: Builden der .NET core Applikation

schlägt der Build aufgrund von fehlender .s2i-Konfiguration (Source-2-Image) fehl. Um diesen Fehler zu beheben, muss Openshift gesagt werden, wo das

Startup-Projekt liegt. Dazu muss ein Ordner und File mit dem Namen `.s2i/environment` erstellt werden. Dies beinhaltet folgendes:

```
1 DOTNET_STARTUP_PROJECT=HelloWorld-netcore/HelloWorld
  -netcore.csproj
```

Wichtig ist weiter zu beachten, dass die .NET Versionen (.NET sowie NuGet-Pakete) mit denjenigen von Openshift Cloud kompatibel sind.

War der Build erfolgreich, muss noch das Deployment konfiguriert werden. Dazu kommt ein weiteres File mit dem Namen `run` ins `.s2i` Verzeichnis. Darin muss die Applikation noch gestartet werden. Dies funktioniert so:

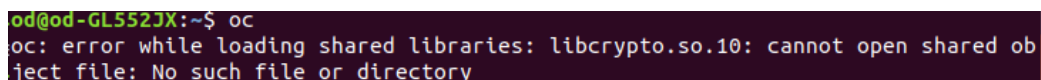
```
1 exec dotnet run
```

Ist auch dieser Schritt vollbracht, kann im Control Panel des Projektes zu Deployments → Routes navigiert werden. Dort erscheint eine Tabelle, wo der Hostname bereits angegeben ist und womit nun die Asp.NET-Core Applikation vom Internet her erreichbar ist.

### 1.1.3 Vorarbeit - Installation der Openshift CLI

Neben dem Web-UI besteht ähnlich wie mit GAE die Möglichkeit, ein Konsolentool für das Konfigurieren der Cloud Projekte zu verwenden. Die Installation des Tools stellte auf Ubuntu 18.04 LTS einige Probleme dar, wie im Folgenden beschrieben wird.

Beim Ausführen der Konsolenapplikation `oc`, auf die in der Doku <sup>1</sup> verwiesen wird, kam der Fehler in Abbildung 6 auf.



```
od@od-GL552JX:~$ oc
oc: error while loading shared libraries: libcrypto.so.10: cannot open shared object file: No such file or directory
```

Figure 6: Fehler beim Ausführen des OC

Gemäss einem Issue auf GitHub<sup>2</sup> handelt es sich um einen fehlerhaften Symlink, der in dieser spezifischen Version enthalten war. Es wird des Weiteren auf eine aktueller Version verwiesen, mit welcher das Problem nicht mehr existiert. Openshift garantiert nur für RHEL<sup>3</sup> Support, weshalb auf anderen Linux Distributionen oder Betriebssystemen mit solchen Fehlern gerechnet werden muss.

Nachdem die Installation des CLI vollbracht ist, können die gleichen Befehle des Web-UIs in der Konsole ausgeführt werden.

<sup>1</sup>[https://docs.openshift.com/online/getting\\_started/basic\\_walkthrough.html](https://docs.openshift.com/online/getting_started/basic_walkthrough.html)

<sup>2</sup><https://github.com/openshift/origin/issues/21061>

<sup>3</sup>Red Hat Enterprise Linux

## Erstellen eines Projektes

```
oc new-project helloWorld --display-name='ASP.NET Core Hello World'
```

## 2 Analyse: OSSM-Definition

Damit sich jemand als Cloud Computing Provider ausgeben kann, sollten folgende Charakteristiken erfüllt sein:

- On-demand
- Self-service
- Scalable
- Measurable

In den folgenden Kapiteln erläutern wir, wie OpenShift diese umsetzt.

### 2.1 On-demand & Self-service

Auf der Startseite findet man eine Katalog aller möglichen Projekttypen. Nach einem Klick auf den gewünschten Projekttyp erscheint ein Dialog, in welchem die spezifische Konfiguration vorgenommen werden kann. Gleichzeitig wird geprüft, ob die definierte Konfiguration plausibel ist. Ist dies der Fall, kann das Projekt erstellt werden. Nach wenigen Sekunden ist das Projekt unter der Rubrik "My Projects" ersichtlich.

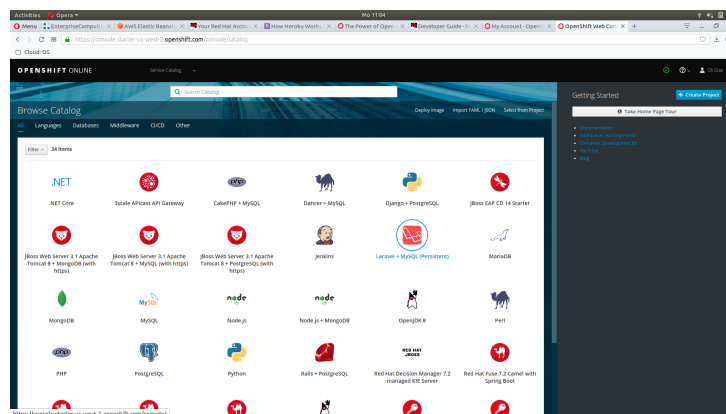


Figure 7: Auswahlkatalog aller möglichen Projekttypen

Die Projektübersicht bietet neben einigen generellen Informationen die Möglichkeit zum Build und Deployment.

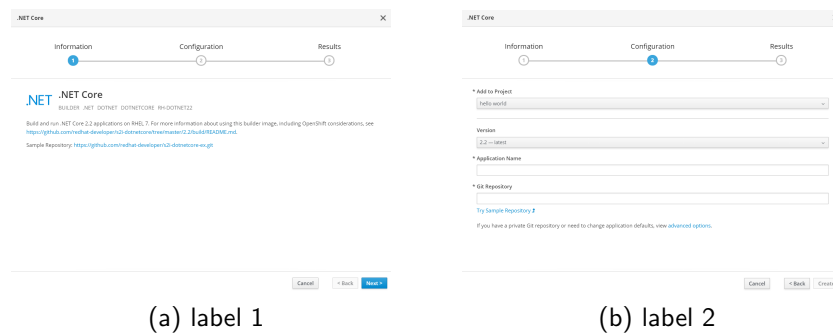


Figure 8: Setup Dialog

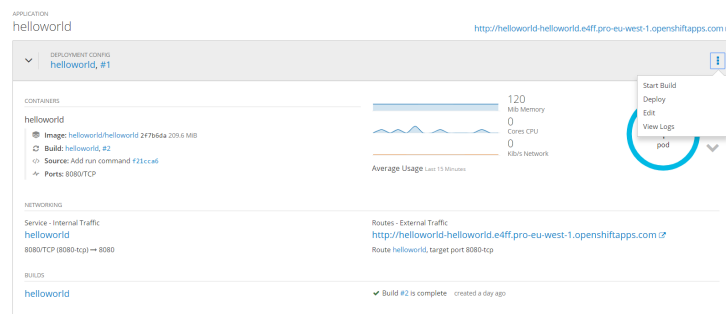


Figure 9: Ansicht der Projektübersicht

## 2.2 Scalable

Die Nutzung von OpenShift erfordert, dass man sich für ein Abonnement entscheidet. Nebst den kostenlosen Einführungsangeboten bedarf die langfristige Nutzung das Abonnement "OpenShift Online Pro". Im Standard bekommt man hierfür folgende Ressourcen:

- 10 Projects
- 2 GB Memory
- 2 GB Terminating Memory
- 2 GB Storage

Um mehr als 10 Projekte zu verwalten bedarf es einem neuen Abonnement. Falls mehr Arbeitsspeicher oder Speicher nötig ist, kann das aktuelle Abonnement angepasst werden, was natürlich einen Einfluss auf den Preis hat. Trotzdem gibt es folgende Begrenzungen:



- Arbeitsspeicher: maximal 48 GB
- Speicher: maximal 150 GB

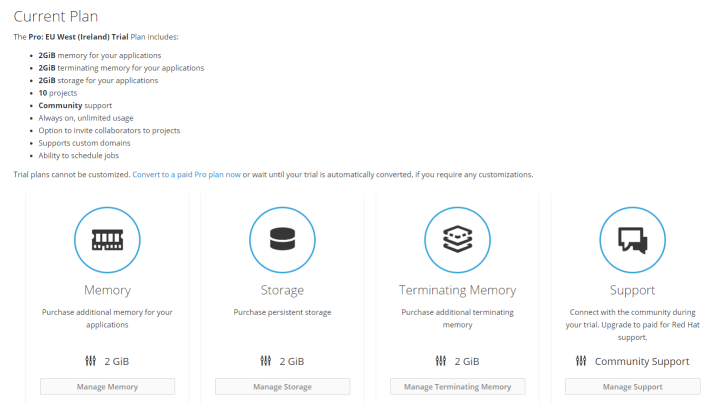


Figure 10: Verwaltung der Ressourcen eines Abonnements

## 2.3 Measurable

Die aktuelle Nutzung der Ressourcen kann lediglich auf Projektstufe eingesehen werden. Diese Übersicht ist ziemlich einfach gehalten, lediglich genutzter Arbeitsspeicher und Speicher werden im Verhältnis zum Maximum angezeigt.

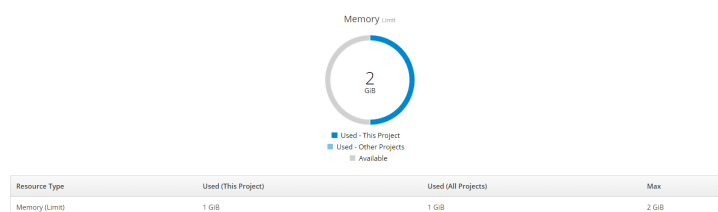


Figure 11: Übersicht der genutzten Ressourcen

## 3 Konzept: Cloud Computing Patterns

Openshift bietet Entwicklern mit dem Konzept PaaS eine Plattform an, auf welcher eine App relative einfach deployed werden kann. Dies bringt den Vorteil, dass der Entwickler sich nicht mit der Komplexität der Building-/Deploying Infrastruktur auseinandersetzen muss. Die Applikation läuft anschliessend auf sogenannten "Pods", welche vergleichbar mit Docker-Container sind. Einerseits

kann die Anzahl Pod's pro Projekt manuell festgelegt und geändert werden. Zudem gibt es die Möglichkeit für Pod Autoscaling. Sobald ein Pod bis zu einem gewissen Grad ausgelastet wird, kommt ein zusätzlicher Pod in Aktion, sofern die maximale Anzahl definierter Pod's nicht erreicht wurde.

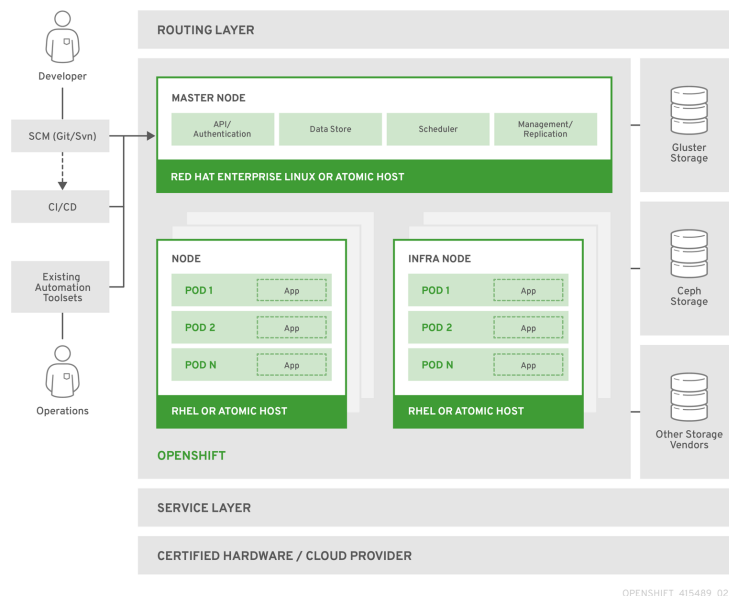


Figure 12: Architekturübersicht der OpenShift-Plattform

## 4 Hands-On: Self Information

## 5 Analyse: Preisrecherche

## 6 Analyse: Preisvergleich eigenes Hosting, IaaS und PaaS

Mit der geschilderten Ausgangslage wurden folgende Annahmen getroffen:

- Beschaffungskosten eigener Server: 1'000
- Server Administrationskosten pro Jahr: 8'000 (10% Stelle mit Ansatz 80'000 Jahresgehalt)

Daraus ergibt sich folgende Kostenaufstellung:

Kostenart	Eigener Server	Google	Amazon E2C	Openshift
Beschaffung	1'000	3'800	3'600	12'100
Betrieb	16'000	4'000	4'000	4'000
Total	17'000	7'800	7'600	16'100

Table 1: Vergleich zwischen lokalen Servern und Cloud Servern

Im Vergleich schneiden die beiden grossen Cloud Provider Amazon und Google am besten ab. Mit etwas Interpretation lässt sich jedoch erkennen, dass die totalen Kosten stark abhängig von der Anzahl Servern ist. Das heisst, je mehr Server für eine Applikation gebraucht werden, umso mehr lohnt sich die Investition in eine eigene Infrastruktur.

Ein weiterer Aspekt stellt die Vertraulichkeit der Applikationsdaten dar. So erwartet der normale Kunde einer Bank, dass seine Kreditkarteninformationen nicht bei einem Server eines externen Anbieters taumeln.

Generell macht es Sinn, Cloud Provider nach den benötigten Funktionalitäten einzuschränken und erst in einem zweiten Schritt preistechnisch zu optimieren. Eine Herausforderung stellt hierbei der Vergleich der einzelnen Provider dar. Oftmals haben unterschiedliche Provider verschieden Angebote mit unterschiedlichen Konfigurationen, was einen Vergleich sinnlos macht.