Lab 3: Ein Docker Image deployen

In diesem Lab werden wir das erste “pre-built” Docker Image deployen und die OpenShift-Konzepte Pod, Service, DeploymentConfig und ImageStream noch etwas genauer anschauen.

Aufgabe: LAB 3.1

Nachdem wir im Lab 2 den Source-to-Image Workflow verwendet haben, um eine Applikation auf OpenShift zu deployen, wenden wir uns nun dem Deployen eines pre-built Docker Images von Docker Hub oder einer anderen Docker-Registry widmen.

Weiterführende Dokumentation

Als ersten Schritt, erstellen wir dazu ein neues Projekt. Ein Projekt ist wie schon in der Einführung beschrieben, eine Gruppierung von Ressourcen (Container, Docker Images, Pods, Services, Routen, Konfiguration, Quotas, Limiten und weiteres). Für das Projekt berechtigte User können diese Ressourcen verwalten. **Achtung: Innerhalb eines OpenShift v3 Clusters muss der Name eines Projektes eindeutig sein.**

Wir erstellen nun daher ein neues Projekt mit dem Namen [DEINNAME]-dockerimage:

```
# oc new-project [DEINNAME]-dockerimage
```

oc new-project wechselt automatisch in das eben neu angelegte Projekt. Mit dem **oc get** Command können Ressourcen von einem bestimmten Typ angezeigt werden.

Man verwendet

```
# oc get project
```

um alle Projekte anzuzeigen, auf die man berechtigt ist.

Sobald das neue Projekt erstellt wurde, können wir in OpenShift mit dem folgenden Befehl das Docker Image deployen:

```
# oc new-app appuio/example-spring-boot
```

```
--> Found Docker image d790313 (3 weeks old) from Docker Hub for
"appuio/example-spring-boot"

    APPUiO Spring Boot App
    ----------------------
    Example Spring Boot App

    Tags: builder, springboot
```

```
    * An image stream will be created as "example-spring-boot:latest" that
will track this image
    * This image will be deployed in deployment config "example-spring-boot"
    * Port 8080/tcp will be load balanced by service "example-spring-boot"
      * Other containers can access this service through the hostname
"example-spring-boot"

--> Creating resources with label app=example-spring-boot ...
    imagestream "example-spring-boot" created
    deploymentconfig "example-spring-boot" created
    service "example-spring-boot" created
--> Success
    Run 'oc status' to view your app.
```

Für das Lab verwenden wir ein APPUiO-Beispiel (Java Spring Boot Applikation):

- Docker Hub: https://hub.docker.com/r/appuio/example-spring-boot/
- GitHub (Source): https://github.com/appuio/example-spring-boot-helloworld

OpenShift legt die nötigen Ressourcen an, lädt das Docker Image in diesem Fall von Docker Hub herunter und deployt anschliessend den ensprechenden Pod.

Tipp: Verwende `oc status` um dir einen Überblick über das Projekt zu verschaffen.

Oder verwende den `oc get pods` Befehl mit dem -w Parameter, um fortlaufend Änderungen an den Ressourcen des Typs Pod anzuzeigen (abbrechen mit ctrl+c):

```
# oc get pods -w
```

Je nach Internetverbindung oder abhängig davon, ob das Image auf dem OpenShift Node bereits heruntergeladen wurde, kann das eine Weile dauern. Schaue dir dabei doch in der Web Console den aktuellen Status des Deployments an:

1. Logge dich in der Web Console ein
2. Wähle das Projekt [DEINNAME]-dockerimage aus
3. Klicke auf Applications
4. Wähle Pods aus

Tipp Um eigene Docker Images für OpenShift zu erstellen, sollte man dabei die folgenden Best Practices befolgen: https://docs.openshift.com/container-platform/3.9/creating_images/guidelines.html

---

Zusatzaufgabe: LAB 3.2

Schaue dir die erstellten Ressourcen mit `oc get [ResourceType] [Name] -o json` und `oc describe [ResourceType] [Name]` aus dem ersten Projekt `[DEINNAME]-example1` an.

Last update: **2018/07/11 12:58**