Table of Contents

1. Overview

In this lab we will be using Red Hat's launcher to learn about cloud native concepts and how they can be applied in Openshit. You will learn how to create an applications on Openshift, build knowledge of health check functionality, scaling, and external configuration.

- Connecting to OpenShift with the client tools
- Create your first OpenShift application
- Openshift UI overview
- The importance of heath checks
- Externalizing your applications configuration
- Running a WAR in OpenShift with JWS

1.1. Prerequisites

- OpenJDK or JDK 1.8 installed
- Access to github and basic git knowledge
- Maven (3.3.x or greater) installed with access to Red Hat repositories
- Access to sites with untrusted certificates
- Ability to install software on workshop laptop. Install client tools allowing you to communicate with openshift
- Participants should have redhat developer accounts and access to the launcher

2. Setting up client tools

In this Lab we will look at how to install the OpenShift CLI tools.

2.1. Installing the Command-line Tools

After completing this section, you should be able to:

- Locate the binaries for the OpenShift Container Platform command-line interface (OCP CLI)
- Install the OCP CLI tools.
- CLI basic configuration

2.2. Locating the binaries

The OCP CLI exposes commands for managing applications, as well as lower-level tools to interact with each component of a system. The binaries for Mac, Windows, and Linux are available for download from the Red Hat Customer Portal via the following methods:

- Supported Binary from the Red Hat Portal: https://access.redhat.com/downloads/content/290

- Upstream Origin Binary: https://github.com/openshift/origin/releases
- Past releases: https://mirror.openshift.com/pub/openshift-v3/clients

2.3. Installing the CLI tools

The CLI is provided as compressed files that can be decompressed to any directory. In order to make it simple for any user to access the OCP CLI, it is recommended that it is made available in a directory mapped to the environment variable called `PATH` from the OS. More information can be found about installation process here: https://docs.openshift.com/container-platform/latest/cli_reference/get_started_cli.html

1. **OSX and Linux:**

   1.1. Copy the binary to the `/usr/local/bin` directory, or one of the paths listed in the `PATH` environment variable.

2. **Windows:**

   2.1. Use oc.exe to open an OpenShift shell. If you getting error from running oc, go to git-scm.com to download git bash for Windows (during installation you need to specify in the selection to integrate with the command prompt)

   2.2. Download and install Notepad++ and install the JSON plugin or use http://jsonlint.com/ to edit and validate JSON.

   http://ammonsonline.com/formatted-json-in-notepad/

   2.3. Configure your default editor to be Notepad++

2.4. CLI basic configuration

The easiest way to initially setup the OpenShift CLI is to use the `oc login` command. It'll interactively ask you a server URL, username and password. The information is automatically saved in a CLI configuration file that is then used for subsequent commands.

To login to a remote server use:

```
$ oc login <hostname>:<port>
```

3. Create an application

In this Lab we will create a basic rest application using the launcher. We will run it locally and on openshift.

3.1. Create an application using the launcher

Create a basic application using Red Hat's launcher wizard

- Create the application using the launcher
- Give your application the name **booster-resthttp-springboot**
- Select build and run locally
- Select **REST API Level 0** in the left column
- Select Spring Boot and the current **RHOAR release** and right column
- Change the **Maven Artifact** in the application information to **booster-resthttp-springboot**
- Click the **Set Up Application** button to download.
- Create a root folder and unzip the application in it
- Inside the folder you will see a README.adoc file follow the instructions in it

NOTE	the application can be found in the 3-create-application directory

4. OpenShit UI

In this Lab we will walk through the openshift UI base on the service you created in the previous step

4.1. Review your pods

- Log into the openshift UI
- Select your project from the list on the right

4.1.1. Builds

- In the **Builds** > **Builds** section, notice the build you created earlier
- In the **Builds** > **Images** section, notice the image you created

4.1.2. Applications

- In the **Applications** > **Services** section notice the service you created. A service serves as an internal load balancer. It identifies a set of replicated pods in order to proxy the connections it receives to them. Backing pods can be added to or removed from a service arbitrarily while the service remains consistently available, enabling anything that depends on the service to refer to it at a consistent address. The default service clusterIP addresses are from the OpenShift Container Platform internal network and they are used to permit pods to access each other.
- In the **Applications** > **Routes** section notice the routes you created. An OpenShift Container Platform route exposes a service at a host name, such as www.example.com, so that external clients can reach it by name.

- Go to the **Overview** section, click on the pod that was created
- Click on the logs tab notice you will see a familar set of logs.
- Click on the terminal tab and run the commands, notice the uber jar is in the container and its process is running

```
$ ls /deployments
$ ps -ef
```

5. Health Check Lab

In this Lab we use an application from the launcher to show the value of the health check.

5.1. Create an application using the launcher

Create a basic application using Red Hat's launcher wizard

- Create the application using the launcher
- Give your application the name **booster-health-check-spr**
- Select build and run locally
- Select **Health Check** in the left column
- Select Spring Boot and the current **RHOAR release** and right column
- Change the **Maven Artifact** in the application information to **booster-health-check-spr**
- Click the **Set Up Application** button to download.
- Create a root folder and unzip the application in it
- Inside the folder you will see a README.adoc file follow the instructions in it

NOTE	the application can be found in the 5-health directory

6. External Configuration Lab

In this Lab we use an application from the launcher to demonstrate how configmaps can be used to externalize configuration

6.1. Create an application using the launcher

Create a basic application using Red Hat's launcher wizard

- Create the application using the launcher
- Give your application the name **booster-configmap-spring-boot**
- Select build and run locally

- Select **Externalized Configuration** in the left column
- Select Spring Boot and the current **RHOAR release** and right column
- Verify the **Maven Artifact** in the application information is set to **booster-configmap-spring-boot**
- Click the **Set Up Application** button to download.
- Create a root folder and unzip the application in it
- Inside the folder you will see a README.adoc file follow the instructions in it

NOTE	the application can be found in the 6-config directory

7. WAR deployment

In this lab we will deploy a WAR into openshift using JWS

7.1. Deploy the Application

- Use the application found in the 7-jws folder
- Follow the instructions in the README.adoc