Templates

1. Connect to OpenShift Cluster

Once you are connected to your client VM, you can log in to the OpenShift cluster.

1. Use the **oc login** command to log in to the cluster, making sure to replace the URL with your cluster’s URL:

oc login -u <Your UserID> <OpenShift master URL>

|  |  |
| --- | --- |
|  | If you see a warning about an insecure certificate, type **y** to connect insecurely. |

**Sample Output**

Login successful.

You don't have any projects. You can try to create a new project, by running

oc new-project <projectname>

2. Install Template

In this section, you build an application and a service with two pods: a front-end web tier and a back-end database tier. This application uses auto-generated parameters and other useful features of OpenShift Container Platform. Note that as part of its JSON configuration, embedded in the source code, this application contains predefined connectivity between the front-end and back-end components. You add resources in a later lab.

This example is, in effect, a quick start—a predefined application that comes in a template and that you can immediately use or customize.

1. Set up the project (replacing **xyz** with your initials):

oc new-project xyz-templates

1. Download the template file and examine it:
2. cd $HOME
3. wget https://raw.githubusercontent.com/redhat-gpte-devopsautomation/openshift3mlbparks/master/mlbparks-template-eap.json

more $HOME/mlbparks-template-eap.json

1. Create a new template:

oc create -f $HOME/mlbparks-template-eap.json

**Sample Output**

template.template.openshift.io/mlbparks-eap created

2.1. Create Instance of Template

Creating a template in a project is also referred to as *uploading* the template. The **mlbparks-template-eap.json** file defines a template. By creating it, you added it to your **xyz-template** project. To make the template available across the cluster, a cluster administrator can add it to the **openshift** namespace by appending the **-n openshift** flag.

1. Use your browser to connect to the OpenShift web console at **https://master.na311.openshift.opentlc.com**.
2. Click C:\Users\681222\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\36A015BF.tmp in the top right corner.
3. Specify the following:
   * **Name**: **xyz-instant-app**
   * **Display Name**: **xyz-instant app example project**
   * **Description**: **A demonstration of an instant app or template**

|  |  |
| --- | --- |
|  | You can also perform this step from the command line:  oc new-project xyz-instant-app --display-name="xyz-instant app example project" --description='A demonstration of an instant-app/template' |

1. Under **My Projects** on the right side of the screen, select **xyz-instant app** example project.
2. Click **Add to Project → Select from Project → xyz-templates**.
3. Select the **mlbparks-eap** application and click **Next**.
   * The idea behind an instant application is that, when you create a template instance, you already have a fully functional application.
   * In this example, your instant application is just a simple web page showing a map of Major League Baseball locations.
4. On the **Information** page, click **Next**.
5. On the **Configuration** page, specify the following options for instantiating the application components:
   * Set the **MONGODB\_PASSWORD** parameter to your favorite password.
   * Scroll down and use **Add Label** to create a label named **version** with the value **1**.
6. Click **Create** to start instantiation of the services, pods, and replication controllers.
   * The **Results** screen is displayed and the build starts immediately.
7. Click **Close**.
   * You can watch the build complete for **xyz-instant app** example project.
8. Wait for the build to finish.
   * You can browse the build logs to follow its progress.

2.2. Use Application

1. After the build is complete, navigate to your application at [http://mlbparks-xyz-instant-app.apps.na311.openshift.opentlc.com](http://mlbparks-xyz-instant-app.apps.na311.openshift.opentlc.com/) (note that your link will be different).
   1. Use HTTP and *not* HTTPS. HTTPS does not work for this example because the form submission was coded with HTTP links.
   2. Make sure the URL matches your particular environment. You can click the route in the web console or enter the following from the command line:
   3. oc project xyz-instant-app

oc get route

**Sample Output**

NAME HOST/PORT PATH SERVICES PORT TERMINATION WILDCARD

mlbparks mlbparks-xyz-instant-app.apps.na311.openshift.opentlc.com mlbparks <all> None

1. Clean up your projects:
   1. Get a list of projects:

oc get projects

**Sample Output**

NAME DISPLAY NAME STATUS

xyz-instant-app xyz-instant app example project Active

xyz-templates Active

* 1. Delete the projects:

oc delete project xyz-instant-app xyz-templates

3. Use Templates and Template Parameters

Quick starts are useful, but there are times when developers want to build components manually. In this exercise, you treat the quick-start example as two separate applications to be wired together.

3.1. Deploy Ephemeral Database Back End

1. Use your browser to connect to the OpenShift web console at **https://master.na311.openshift.opentlc.com**.
2. Create a project for the database back end:
   1. Click C:\Users\681222\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\82606525.tmp in the top right corner.
   2. Specify the following:
      * **Name**: **xyz-test-templates**
      * **Display Name**: **Templates Testing Project**
      * **Description**: **Project for testing templates**

|  |  |
| --- | --- |
|  | You can also perform this step from the command line:  oc new-project xyz-test-templates --display-name="Templates Testing Project" --description='Project used to test templates' |

1. Deploy an ephemeral MongoDB database:
   1. From the **xyz-test-templates** project’s **Overview** screen, click **Add to project** and select **Browse Catalog**.
   2. Search for "mongodb ephemeral", then select **MongoDB Ephemeral**.

|  |  |
| --- | --- |
|  | Be careful not to select other similarly named options with "MongoDB Ephemeral" in their names, such as "Node.js + MongoDB Ephemeral". |

* 1. On the **Configuration** page, set the following template parameters:
     + **Add to Project**: **xyz-test-templates**
     + **Database Service Name**: **mongodb**
     + **MongoDB Connection Username**: **mongouser**
     + **MongoDB Connection Password**: **redhat**
     + **MongoDB Database Name**: **mydb**

|  |  |
| --- | --- |
|  | Make sure you set these values correctly; otherwise the application does not connect to this database back end. |

* 1. Click **Next**.
  2. Select **Create a secret in Templates Testing Project to be used later**.
  3. Click **Create**, and then click the **Continue to the project overview** link.

|  |  |
| --- | --- |
|  | You can also create the template instance from the command line:  oc new-app --template=mongodb-ephemeral --name=mongdb --param MONGODB\_USER=mongouser --param MONGODB\_PASSWORD=redhat --param MONGODB\_DATABASE=mydb --param DATABASE\_SERVICE\_NAME=mongodb |

3.2. Examine Database Back End

1. Switch to the templates project:
   1. Get a list of projects:

oc get projects

**Sample Output**

NAME DISPLAY NAME STATUS

xyz-test-templates Templates Testing Project Active

* 1. Switch to the project:

oc project xyz-test-templates

**Sample Output**

Now using project "xyz-test-templates" on server "https://master.na311.openshift.opentlc.com:443"

1. Examine the objects that were created as part of the **mongodb-ephemeral** template:
   1. Get the deployment configuration name:

oc get dc

**Sample Output**

NAME REVISION DESIRED CURRENT TRIGGERED BY

mongodb 1 1 1 config,image(mongodb:3.2)

* 1. Get the service name:

oc get service

**Sample Output**

NAME CLUSTER-IP EXTERNAL-IP PORT(S) AGE

mongodb 172.30.33.154 <none> 27017/TCP 36s

* + - A deployment configuration is available for your instance. The service name is the same as that of your **DATABASE\_SERVICE\_NAME** parameter.

1. Verify that the values of the environment variables in the deployment configuration (**dc**) are correct:
   1. List the environmental variables:

oc set env dc mongodb --list

**Sample Output**

*# deploymentconfigs/mongodb, container mongodb*

*# MONGODB\_USER from secret mongodb, key database-user*

*# MONGODB\_PASSWORD from secret mongodb, key database-password*

*# MONGODB\_ADMIN\_PASSWORD from secret mongodb, key database-admin-password*

*# MONGODB\_DATABASE from secret mongodb, key database-name*

* 1. Get the user and password stored in the **mongodb** secret:

oc get secret mongodb

**Sample Output**

NAME TYPE DATA AGE

mongodb Opaque 4 2m

* 1. Get the details:

oc get secret mongodb -o yaml

**Sample Output**

apiVersion: v1

data:

database-admin-password: TGJtb1JLeVNRSUtKS2J5bw==

database-name: bXlkYg==

database-password: cmVkaGF0

database-user: bW9uZ291c2Vy

kind: Secret

metadata:

annotations:

template.openshift.io/expose-admin\_password: '{.data[''database-admin-password'']}'

template.openshift.io/expose-database\_name: '{.data[''database-name'']}'

template.openshift.io/expose-password: '{.data[''database-password'']}'

template.openshift.io/expose-username: '{.data[''database-user'']}'

creationTimestamp: 2019-01-28T17:10:26Z

labels:

template: mongodb-ephemeral-template

template.openshift.io/template-instance-owner: 9af0bfe1-231f-11e9-acf4-025f6b8bf8ee

name: mongodb

namespace: xyz-test-templates

resourceVersion: "17025472"

selfLink: /api/v1/namespaces/xyz-test-templates/secrets/mongodb

uid: 9af374bf-231f-11e9-acf4-025f6b8bf8ee

type: Opaque

* + - Secret values are encoded in base64 format.

1. Pass the content to the **base64 -d** command to decode it:
   1. Pass the user:

oc get secret mongodb --template '{{ index .data "database-user"}}'|base64 -d

**Sample Output**

mongouser

* 1. Pass the password:

oc get secret mongodb --template '{{ index .data "database-password"}}'|base64 -d

**Sample Output**

redhat

3.3. Deploy Application’s JBoss EAP Front End

1. Create an application with the <https://github.com/redhat-gpte-devopsautomation/openshift3mlbparks> Git repository:

oc new-app --image-stream="openshift/jboss-eap71-openshift:1.3" https://github.com/redhat-gpte-devopsautomation/openshift3mlbparks MONGODB\_USER=mongouser MONGODB\_PASSWORD=redhat MONGODB\_DATABASE=mydb DATABASE\_SERVICE\_NAME=mongodb

1. Verify that your service is in place:

oc get service

**Sample Output**

NAME CLUSTER-IP EXTERNAL-IP PORT(S) AGE

mongodb 172.30.33.154 <none> 27017/TCP 7m

openshift3mlbparks 172.30.74.42 <none> 8080/TCP,8443/TCP,8778/TCP 3m

1. Create an external route to your front-end application:
   1. Expose the service:

oc expose service openshift3mlbparks

**Sample Output**

route.route.openshift.io/openshift3mlbparks exposed

* 1. Get the route:

oc get route

NAME HOST/PORT PATH SERVICES PORT TERMINATION WILDCARD

openshift3mlbparks openshift3mlbparks-xyz-test-templates.apps.na311.openshift.opentlc.com openshift3mlbparks 8080-tcp None

* + - If you do not specify a host name, the default subdomain route creates the route.

1. Wait for the build to complete, then test your environment:

oc logs -f bc/openshift3mlbparks

**Sample Output**

[...]

[INFO] ------------------------------------------------------------------------

[INFO] BUILD SUCCESS

[INFO] ------------------------------------------------------------------------

[INFO] Total time: 43.695 s

[INFO] Finished at: 2019-01-28T17:19:30Z

[INFO] Final Memory: 27M/179M

[INFO] ------------------------------------------------------------------------

Copying all war artifacts from /tmp/src/target directory into /opt/eap/standalone/deployments for later deployment...

Copying all ear artifacts from /tmp/src/target directory into /opt/eap/standalone/deployments for later deployment...

Copying all rar artifacts from /tmp/src/target directory into /opt/eap/standalone/deployments for later deployment...

Copying all jar artifacts from /tmp/src/target directory into /opt/eap/standalone/deployments for later deployment...

Copying all war artifacts from /tmp/src/deployments directory into /opt/eap/standalone/deployments for later deployment...

'/tmp/src/deployments/ROOT.war' -> '/opt/eap/standalone/deployments/ROOT.war'

Copying all ear artifacts from /tmp/src/deployments directory into /opt/eap/standalone/deployments for later deployment...

Copying all rar artifacts from /tmp/src/deployments directory into /opt/eap/standalone/deployments for later deployment...

Copying all jar artifacts from /tmp/src/deployments directory into /opt/eap/standalone/deployments for later deployment...

Pushing image docker-registry.default.svc:5000/xyz-test-templates/openshift3mlbparks:latest ...

Pushed 0/7 layers, 5% complete

Pushed 1/7 layers, 17% complete

Pushed 2/7 layers, 29% complete

Pushed 2/7 layers, 41% complete

Pushed 3/7 layers, 43% complete

Push successful

1. Wait for the pods to start, then verify that your application is running and connecting to the database:

http://openshift3mlbparks-xyz-test-templates.apps.na311.openshift.opentlc.com

4. Clean Up Environment

1. To clean up this project, simply delete the project:

oc delete project xyz-test-templates

Build Version: 1.9R : Last updated 2019-03-25 17:06:59 EDT