Guided Exercise: Creating an Application with the Web Console

In this exercise, you will create, build, and deploy an application to an OpenShift cluster using the OpenShift web console.

**Outcomes**

You should be able to create, build, and deploy an application to an OpenShift cluster using the web console.

Get the lab files by executing the lab script:

**[student@workstation ~]$ lab openshift-webconsole start**

The lab script verifies that the OpenShift cluster is running.

1. Inspect the PHP source code for the sample application and create and push a new branch named console to use during this exercise.
   1. Enter your local clone of the DO180-apps Git repository and checkout the master branch of the course's repository to ensure you start this exercise from a known good state:
   2. **[student@workstation ~]$ cd ~/DO180-apps**
   3. **[student@workstation DO180-apps]$ git checkout master**

*...output omitted...*

* 1. Create a new branch to save any changes you make during this exercise:
  2. **[student@workstation DO180-apps]$ git checkout -b console**
  3. Switched to a new branch 'console'
  4. **[student@workstation DO180-apps]$ git push -u origin console**
  5. *...output omitted...*
  6. \* [new branch] console -> console

Branch 'console' set up to track remote branch 'console' from 'origin'.

* 1. Review the PHP source code of the application, inside the the php-helloworld folder.

Open the index.php file in the /home/student/DO180-apps/php-helloworld folder:

<?php

print "Hello, World! php version is " . PHP\_VERSION . "\n";

?>

The application implements a simple response which returns the PHP version it is running.

1. Open a web browser and navigate to https://console-openshift-console.${RHT\_OCP4\_WILDCARD\_DOMAIN} to access the OpenShift web console. Log in and create a new project named *youruser*-console.
   1. Load your classroom environment configuration.

Run the following command to load the environment variables created in the first guided exercise:

**[student@workstation ~]$ source /usr/local/etc/ocp4.config**

* 1. Retrieve the value of the wildcard domain specific to your cluster, using the $RHT\_OCP4\_WILDCARD\_DOMAIN
  2. **[student@workstation ~]$ echo $RHT\_OCP4\_WILDCARD\_DOMAIN**
  3. apps.***cluster.lab.example.com***
  4. Open the Firefox browser and navigate to https://console-openshift-console.${RHT\_OCP4\_WILDCARD\_DOMAIN} to access the OpenShift web console. Log in to the OpenShift console using your credentials.
  5. Create a new project named ***youruser*-console**. You can type any values you prefer in the other fields.

|  |
| --- |
|  |

* 1. Figure 6.13: Create a new project

|  |
| --- |
|  |

* 1. Figure 6.14: Create a new project
  2. After you have completed the required fields, click **Create** in the Create Project dialog box to go to the Project Status page for the *youruser*-console project:

|  |
| --- |
|  |

* 1. Figure 6.15: Project Status page

1. Create the new php-helloworld application with a PHP template.
   1. Switch to the **Developer** perspective using the drop-down at the top of the left-hand menu:

|  |
| --- |
|  |

* 1. Figure 6.16: Developer perspective drop-down
  2. Click **From Catalog** to display a list of technology templates. Uncheck the **Operator Backed** checkbox to see all of the templates.

|  |
| --- |
|  |

* 1. Figure 6.17: Developer Catalog page
  2. Enter php in the **Filter by keyword** field.

|  |
| --- |
|  |

* 1. Figure 6.18: Finding PHP-related templates
  2. After filtering, click the PHP builder image to display the PHP dialog box. Click **Create Application**to display the **Create Source-to-Image Application** page.

|  |
| --- |
|  |

* 1. Figure 6.19: Configuring Source-to-Image for a PHP application
  2. Change the **Builder Image Version** to PHP version 7.3.

Specify the location of the source code git repository: https://github.com/*yourgituser*/DO180-apps.

Use the **Advanced Git Options** to set the context directory to php-helloworld and branch console for this exercise

|  |
| --- |
|  |

Figure 6.20: Setting Advanced Git Options for the application

Enter **php-helloworld** for both the application name and the name used for associated resources. Select **Deployment Config** as the resource type.

|  |
| --- |
|  |

Figure 6.21: Setting application options

Scroll to the bottom of the page, and select **Create a route to the application**. Click **Create** to create the required OpenShift and Kubernetes resources for the application.

You are redirected to the Topology page:

|  |
| --- |
|  |

Figure 6.22: Topology page

This page indicates that the php-helloworld application is created. The DC annotation to the left of the php-helloworld link is an acronym for Deployment Config. This link redirects to a page containing information about the application's deployment configuration.

* 1. Switch back to the **Administrator** perspective for the remainder of the exercise:

|  |
| --- |
|  |

* 1. Figure 6.23: Administrator perspective drop-down

1. Use the navigation bar on the left side of the OpenShift web console to locate information for the application's OpenShift and Kubernetes resources:
   1. DeploymentConfig
   2. BuildConfig
   3. Build Logs
   4. Service
   5. Route
   6. Examine the deployment configuration. In the navigation bar, click **Workloads** to reveal more menu choices. Click **Deployment Configs** to display a list of deployment configurations for the *youruser*-console project. Click the php-helloworld link to display deployment configuration information.

|  |
| --- |
|  |

* 1. Figure 6.24: Application deployment configuration details page
  2. Explore the available information from the **Details** tab. The build may still be running when you reach this page, so the DC might not have a value of 1 pod, yet.
  3. If you click the up and down arrow icons next to the doughnut chart that indicates the number of pods, you can scale the application up and down horizontally.
  4. Examine the build configuration. In the navigation bar, click **Builds** to reveal more menu choices. Click **Build Configs** to display a list of build configurations for the *youruser*-console project. Click the php-helloworld link to display the build configuration for the application.

|  |
| --- |
|  |

* 1. Figure 6.25: Application build configuration details page
  2. Explore the available information from the **Details** tab. The **YAML** tab allows you to view and edit the build configuration as a YAML file. The **Builds** tab provides an historical list of builds, along with a link to more information for each build. The **Environment** tab allows you to view and edit environment variables for the application's build environment. The **Events** tab displays a list of build related events and metadata.
  3. Examine the logs for the Source-to-Image build of the application. In the **Builds** menu, click **Builds** to display a list of recent builds for the *youruser*-console project.

Click the php-helloworld-1 link to access information for the first build of the php-helloworld application:

|  |
| --- |
|  |

Figure 6.26: An application build details page

Explore the available information from the **Details** tab. Next, click the **Logs** tab. A scrollable text box contains output from the build process:

|  |
| --- |
|  |

Figure 6.27: Logs for an application build

When Podman builds a container image, similar output is observed compared with the output shown in the browser.

* 1. Locate information for the php-helloworld application's service. In the navigation bar, click **Networking** to reveal more menu choices. Click **Services** to display a list of services for the *youruser*-console project. Click the php-helloworld link to display the information associated with the application's service:

|  |
| --- |
|  |

* 1. Figure 6.28: Service details page
  2. Explore the available information from the **Details** tab. The **YAML** tab allows you to view and edit the service configuration, as a YAML file. The **Pods** tab displays the current list of pods that provide the application service.
  3. Locate external route information for the application. On the navigation bar, click **Networking** → **Routes** to display a list of configured routes for the *youruser*-console project. Click the php-helloworld link to display information associated with the application's route:

|  |
| --- |
|  |

* 1. Figure 6.29: Route details page
  2. Explore the available information from the **Details** tab. The **Location** field provides a link to the external route for the application; http://php-helloworld-${RHT\_OCP4\_DEV\_USER}-console.${RHT\_OCP4\_WILDCARD\_DOMAIN}. Click the link to access the application in a new tab:

|  |
| --- |
|  |

* 1. Figure 6.30: Initial PHP application results

1. Modify the application code, commit the change, push the code to the remote Git repository, and trigger a new application build.
   1. Enter the source code directory:

**[student@workstation DO180-apps]$ cd ~/DO180-apps/php-helloworld**

* 1. Add the second print line statement in the index.php page to read "A change is in the air!" and save the file. Add the change to the Git index, commit the change, and push the changes to the remote Git repository.
  2. **[student@workstation php-helloworld]$ vim index.php**
  3. **[student@workstation php-helloworld]$ cat index.php**
  4. <?php
  5. print "Hello, World! php version is " . PHP\_VERSION . "\n";
  6. **print "A change is in the air!\n";**
  7. ?>
  8. **[student@workstation php-helloworld]$ git add index.php**
  9. **[student@workstation php-helloworld]$ git commit -m 'updated app'**
  10. [console d198fb5] updated app
  11. *...output omitted...*
  12. 1 file changed, 1 insertion(+), 1 deletion(-)
  13. **[student@workstation php-helloworld]$ git push origin console**
  14. Counting objects: 7, done.
  15. Delta compression using up to 2 threads.
  16. Compressing objects: 100% (3/3), done.
  17. Writing objects: 100% (4/4), 409 bytes | 0 bytes/s, done.
  18. Total 4 (delta 1), reused 0 (delta 0)
  19. *...output omitted...*
  20. Trigger an application build manually from the web console.

On the navigation bar, click **Builds** → **Build Configs** and then click the php-helloworld link to access the Build Config Details page. From the **Actions** menu in the upper right of the screen, click **Start Build**:

|  |
| --- |
|  |

Figure 6.31: Start an application build

You are redirected to a Build Details page for the new build. Click the **Logs** tab to monitor progress of the build. The last line of a successful build contains Push successful.

When the build completes, the deploy starts. Go to the **Workloads** → **Pods** section, and wait for the new pod is deployed and running.

* 1. Reload the http://php-helloworld-${RHT\_OCP4\_DEV\_USER}-console.${RHT\_OCP4\_WILDCARD\_DOMAIN} URL in the browser. The application response corresponds to the updated source code:

|  |
| --- |
|  |

* 1. Figure 6.32: Updated web application output

1. Delete the project. On the navigation bar, click **Home** → **Projects**. Click the icon at the right side of the row containing an entry for the *youruser*-console project. Click **Delete Project** from the menu that appears.

|  |
| --- |
|  |

1. Figure 6.33: Delete a project
2. Enter *youruser*-console in the confirmation dialog box, and click **Delete**.

**Finish**

On workstation, run the **lab openshift-webconsole finish** script to complete this lab.

**[student@workstation php-helloworld]$ lab openshift-webconsole finish**

This concludes the guided exercise.