

## Table of Contents

---

### Creating Applications Lab

1. Deploy Application on Web Console
    - 1.1. Connect To and Explore Web Console
    - 1.2. Create New Project
    - 1.3. Create New Application
    - 1.4. Scale Deployment and Topology View
  2. Customize Build Script
    - 2.1. Clone Repository and Launch Application from Local Copy
    - 2.2. Fork Repository
    - 2.3. Add Script to Repository
    - 2.4. Create Application From Repository With Custom Build Script
- 

## Creating Applications Lab

---

This lab includes the following sections:

- **Deploy Application on Web Console**

In this section, you deploy an application from a code repository and follow the build logs on the OpenShift Enterprise web console and CLI.

- **Customize Build Script**

- Create an application from a forked Git repository, inject a custom build script, and start a rebuild from the web console.
  - Review your custom script messages in the logs.
- 

## 1. Deploy Application on Web Console

---

Here, you connect to and become familiar with the web console, create a project and an application, and scale a deployment and the topology view.

### 1.1. Connect To and Explore Web Console

---

1. Use your browser to go to the OpenShift web console at `https://master00-GUID.oslab.opentlc.com:8443`.



If you have not done so already, accept the untrusted certificate.

2. Log in as `andrew` with the password `r3dh4t1!`.
3. Take a few minutes to browse your projects.

### 1.2. Create New Project

---

1. Click **Projects** and select **View all projects** to return to the Projects view.
2. Click the blue **New Project** button in the top right corner.
3. Give the new project a name, display name, and description:
  - **Name:** `my-ruby-project`
  - **Display Name:** `My Ruby Example Project`
  - **Description:** An explanation of your choice

Once the project is in place, the **Select Image or Template** screen is displayed.

### 1.3. Create New Application

---

1. In the **Select Image or Template** screen, type `ruby` in the search field to filter the available instant apps, templates, and


builder images.

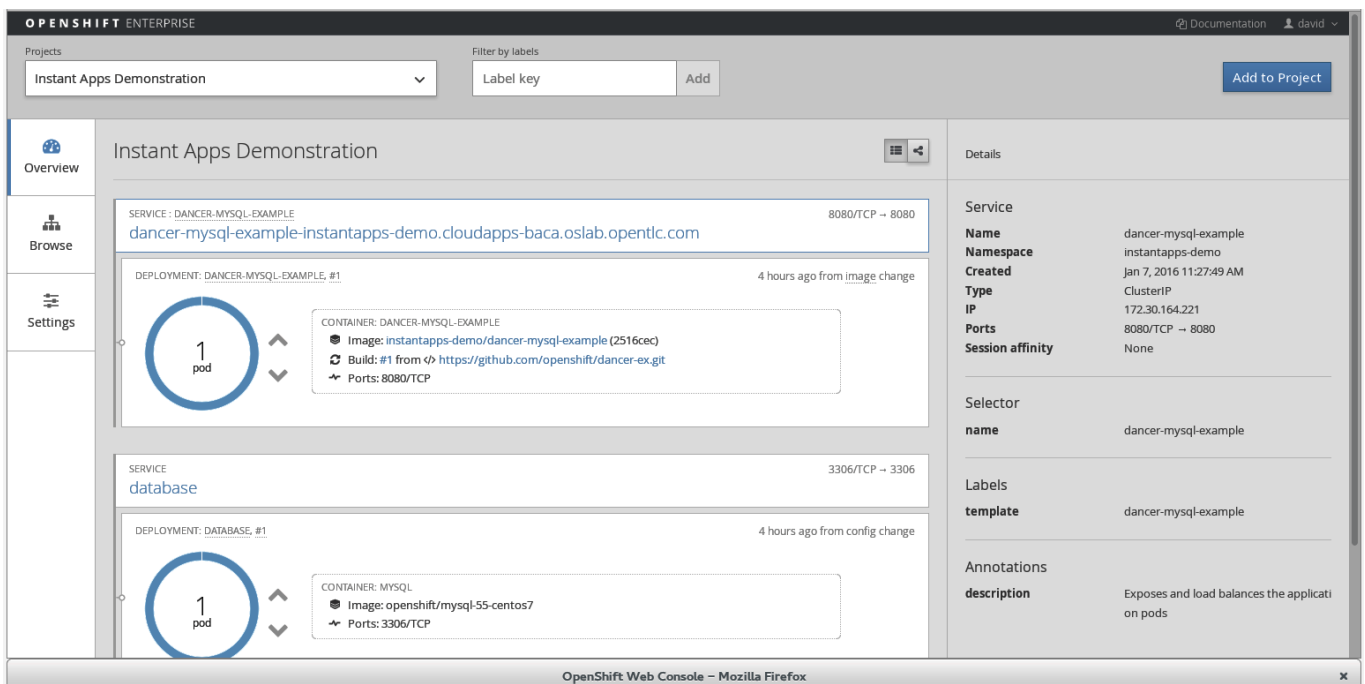
2. Select the **ruby:2.0** builder image on the right.
3. Specify the name and Git repository URL:
  - **Name:** **my-ruby-hello-world**.
  - **Git Repository URL:** **https://github.com/openshift/ruby-hello-world**.
4. Click **Show advanced build and deployment options** and select the following options:
  - a. Decide if you want a route for your application.
  - b. Optionally, define the triggers and environment variables for the deployment.
  - c. Change the scaling parameter to 3.
  - d. Create a label for **environment** by typing **dev**.
5. Accept and start the application.
6. Review the information that is displayed.
7. Click **Continue to Overview** to go to the application's **Overview** screen.
8. Click **View Logs** to verify that a build is in progress.
9. Review the log as the build progresses.
10. Wait for the build to complete and use a browser to navigate to the application route: <http://my-ruby-hello-world-my-ruby-project.cloudapps-GUID.oslab.opentlc.com>
  - a. The database for our application isn't running, so expect to see the web page mention that.




- You can also use the command line to create a new application:  
**oc new-app https://github.com/openshift/ruby-hello-world -l environment=dev**.
- To change scaling from the command line, use **oc scale**.


## 1.4. Scale Deployment and Topology View

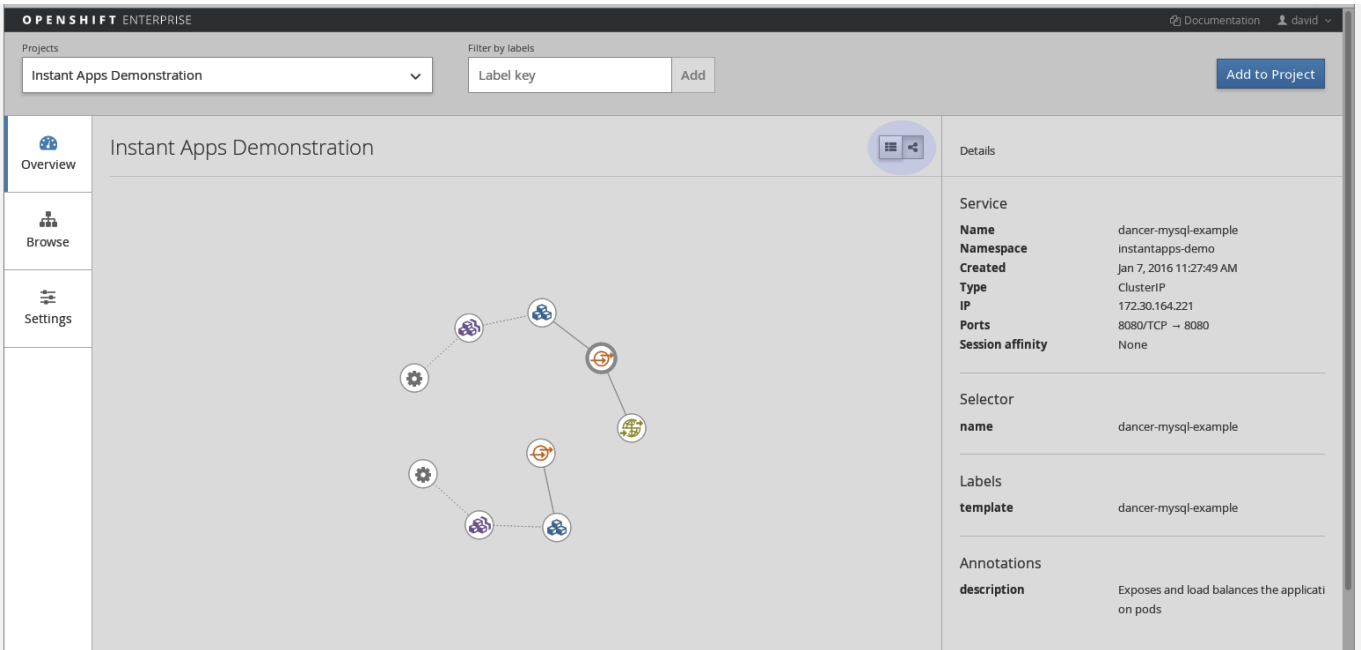
1. Go back to your application's **Overview** screen by clicking **Overview** at the upper left side.
2. Observe the circle that shows the current number of pods, which is 1. You can increase that number by clicking the  button next to it.



The screenshot shows the OpenShift Enterprise Web Console interface. At the top, there's a header with 'OPENSHIFT ENTERPRISE' and user information. Below the header, there's a navigation bar with 'Overview', 'Browse', and 'Settings' tabs. The main content area is titled 'Instant Apps Demonstration'. It shows two services: 'dancer-mysql-example' and 'database'. Each service has a deployment with a pod count of 1. The 'dancer-mysql-example' service is running on port 8080/TCP, and the 'database' service is running on port 3306/TCP. The console also shows details for each service, including name, namespace, created time, type, IP, ports, and session affinity.

1. Click the  button twice to increase the number of replicas to 3.
2. Hover over **Browse** and select **Pods** to take a look at your new pods.

- Go back to your application's **Overview** screen by clicking **Overview** again.
  - The current and default view of the **Overview** screen is the **Tile View**.
  - To switch to the **Topology View**, click the button on the upper right hand side that looks like the  character or a social-media Share button, which it is not.



The screenshot shows the OpenShift Enterprise interface. At the top, there's a header with 'OPENSHIFT ENTERPRISE' and user information. Below that, a navigation bar includes 'Projects' (set to 'Instant Apps Demonstration') and a 'Filter by labels' section. The main content area is titled 'Instant Apps Demonstration' and displays a topology diagram of services and pods. A button in the top right corner, which looks like a share button, is highlighted with a blue circle. The right sidebar shows details for the 'dancer-mysql-example' service, including its name, namespace, creation time, type, IP, ports, and session affinity.

- Click the objects and consider their relationships.

## 2. Customize Build Script

OpenShift Enterprise 3 supports customization of both the build and run processes. Generally speaking, this involves modifying the S2I scripts from the builder image. While building your code, OpenShift Enterprise checks the scripts in your repository's `.sti/bin` folder to see if they override or supersede the builder image's scripts. If it finds scripts that do so, it executes those scripts.

For details on the scripts and their execution and customization, go to

[https://docs.openshift.com/enterprise/3.1/creating\\_images](https://docs.openshift.com/enterprise/3.1/creating_images).

### 2.1. Clone Repository and Launch Application from Local Copy

- Log in to OpenShift Enterprise as `marina`:
  - Connect to the OpenShift Enterprise master by following the same steps as before.
  - When prompted, type the username and password:
    - **Username:** `marina`
    - **Password:** `r3dh4t1!`

```
[root@master00-GUID ~]# su - marina
[marina@master00-GUID ~]$ oc login -u marina --insecure-skip-tls-verify --server=https://master00-
${guid}.oslab.opentlc.com:8443

[marina@master00-GUID ~]$ oc new-project custom-s2i-script --display-name="Custom S2I Build Script" \
--description="This is the project we use to learn how to create a customized build script"
```

### 2.2. Fork Repository



This section requires a GitHub account. Create one if you do not have one already. It is free and useful.

1. From the GitHub web UI, fork the <https://github.com/openshift/ruby-hello-world> Git repository into your own Git account by clicking **Fork** in the upper right corner.
  - This creates a repository in your Git account with a name similar to <https://github.com/yourname/ruby-hello-world/>, where *yourname* is your Git username.
2. Clone this <https://github.com/yourname/ruby-hello-world> repository so that you can edit it locally and test a Red Hat-customized script with it:



Be sure to replace *yourname* with your Git username.

```
[marina@master00-GUID ~]$ git clone https://github.com/yourname/ruby-hello-world
Cloning into 'ruby-hello-world'...
remote: Counting objects: 249, done.
remote: Total 249 (delta 0), reused 0 (delta 0), pack-reused 249
Receiving objects: 100% (249/249), 36.79 KiB | 0 bytes/s, done.
Resolving deltas: 100% (86/86), done.
```

3. Create an application by running **oc new-app** in the local repository:

```
[marina@master00-GUID ~]$ cd ruby-hello-world/
[marina@master00-GUID ruby-hello-world]$ oc new-app . --docker-image=registry.access.redhat.com/openshift3/ruby-20-rhel7
```

4. View the current build status and build logs:

```
[marina@master00-GUID]$ oc get builds
NAME                TYPE      FROM      STATUS      STARTED          DURATION
ruby-hello-world-1  Docker    Git@master Running     9 seconds ago   9s
```

5. View the build log:

```
[marina@master00-GUID ~]$ oc logs -f builds/ruby-hello-world-1
...
... Omitted Output ...
...
Removing intermediate container 049a12eb5ca5
Successfully built 995028e8bee2
I1127 02:41:37.640510      1 docker.go:86] Pushing image 172.30.42.118:5000/custom-s2i-script/ruby-hello-world:latest ...
I1127 02:44:25.867627      1 docker.go:90] Push successful
```

6. Verify that your pod has deployed:

```
[marina@master00-GUID ~]$ oc get pods
NAME                READY   STATUS    RESTARTS   AGE
ruby-hello-world-1-70mlb  1/1     Running   0          12s
ruby-hello-world-1-build  0/1     Completed 0          9m
```

## 2.3. Add Script to Repository

1. Open a new tab in your browser, go to [http://www.opentlc.com/download/ose\\_implementation/resources/3.1/assemble](http://www.opentlc.com/download/ose_implementation/resources/3.1/assemble), and copy all of the text there.
2. Go to your GitHub repository for your application from the previous section.
3. In the GitHub web UI, navigate to the **.sti/bin** folder.
4. Click the **+** button at the top (to the right of **bin** in the breadcrumb).
5. Name your file **assemble**.

6. In the GitHub web UI, paste the content you copied earlier into the text area.
7. Type a commit message in the text field.
8. Click **Commit**.

## 2.4. Create Application From Repository With Custom Build Script

1. From your browser, go to the OpenShift web console at `https://master00-GUID.oslab.opentlc.com:8443`.



If prompted, accept the untrusted certificate.

2. Log in as `marina` with the password `r3dh4t1!`.
3. Click the blue **New Project** button in the top right corner.
4. Specify the project name, display name, and description:
  - **Name:** `my-custom`
  - **Display Name:** `My custom assemble script project`
  - **Description:** An explanation of your choice
    - Once the project is in place, the **Select Image or Template** screen is displayed.
5. In the **Select Image or Template** screen, type `ruby` in the search field to filter the available Instant Apps, Templates, and Builder Images.
6. Select the `ruby:2.0` builder image from the right hand side.
7. Specify the name and Git repository URL:
  - **Name:** `my-custom-builder-test`
  - **Git Repository URL:** `https://github.com/yourname/ruby-hello-world`



Remember to replace *yourname* with your Git username in the above command.

8. Follow the build process logs and watch for this custom assemble script message, which confirms that the custom script ran:

```
2015-04-27T22:23:24.110630393Z --> CUSTOM S2I ASSEMBLE COMPLETE
```