Guided Exercise: Creating Containers with Podman

Create several Podman containers by using different options.

Outcomes

You should be able to run containers in Podman by using the `podman run` command.

As the `student` user on the `workstation` machine, use the `lab` command to prepare your system for this exercise.

```
[student@workstation ~]$ lab start basics-creating
```

Instructions

1. Use the `registry.ocp4.example.com:8443/ubi9/ubi-minimal:9.5` image to create a new container that prints the `Hello Red Hat` text.

   Use the `podman pull` command to fetch the image from the registry.

   ```
   [student@workstation ~]$ podman pull \
     registry.ocp4.example.com:8443/ubi9/ubi-minimal:9.5
   Trying to pull registry.ocp4.example.com:8443/ubi9/ubi-minimal:9.5...
   ...output omitted...
   Writing manifest to image destination
   Storing signatures
   a0c2...eca6
   ```

   Use the `podman images` command to verify that the image is available locally.

   ```
   [student@workstation ~]$ podman images
   REPOSITORY                                      TAG     IMAGE ID...
   ...output omitted...
   registry.ocp4.example.com:8443/ubi9/ubi-minimal  9.5   a0c247...
   ```

   In a command line terminal, use `podman run` to create a container. Provide an `echo` command to be executed inside the container.

   ```
   [student@workstation ~]$ podman run --rm \
     registry.ocp4.example.com:8443/ubi9/ubi-minimal:9.5 cat /etc/os-release
   NAME="Red Hat Enterprise Linux"
   VERSION="9.5 (Plow)"
   ID="rhel"
   ...output omitted...
   ```

   Verify that the container is not running after the execution finishes.

   ```
   [student@workstation ~]$ podman ps
   CONTAINER ID  IMAGE       COMMAND     CREATED     STATUS      PORTS       NAMES
   ```

2. Explore setting and printing environment variables by using the `registry.ocp4.example.com:8443/ubi9/ubi-minimal:9.5` container image.

   Use the `-e` option in the `podman run` command to set environment variables. Use the `printenv` command that is packaged in the container image to print the values of the environment variables.

   ```
   [student@workstation ~]$ podman run --rm -e GREET=Hello -e NAME='Red Hat' \
     registry.ocp4.example.com:8443/ubi9/ubi-minimal:9.5 printenv GREET NAME
   Hello
   Red Hat
   ```

   Verify that the container is not running after the execution finishes.

   ```
   [student@workstation ~]$ podman ps
   CONTAINER ID  IMAGE       COMMAND     CREATED     STATUS      PORTS       NAMES
   ```

3. By using the `registry.ocp4.example.com:8443/ubi9/httpd-24` image, run a new container that creates an Apache HTTP server.

   Use the `podman run` command to pull the container image and start the `httpd` container. Use the `-p` option in the `podman run` command to redirect traffic from port `8080` on your machine to port `8080` inside the container.

```
[student@workstation ~]$ podman run --rm -p 8080:8080 \
  registry.ocp4.example.com:8443/ubi9/httpd-24
...output omitted...
[Thu Apr 21 12:58:57.048491 2022] [ssl:warn] [pid 1:tid 140257793613248] AH01909: 10.0.2.100:8443:0 server certificate doe
s NOT include an ID which matches the server name
[Thu Apr 21 12:58:57.048899 2022] [:notice] [pid 1:tid 140257793613248] ModSecurity for Apache/2.9.2 (http://www.modsecuri
ty.org/) configured.
...output omitted...
[Thu Apr 21 12:58:57.136272 2022] [mpm_event:notice] [pid 1:tid 140257793613248] AH00489: Apache/2.4.37 (Red Hat Enterpris
e Linux) OpenSSL/1.1.1k configured -- resuming normal operations
[Thu Apr 21 12:58:57.136332 2022] [core:notice] [pid 1:tid 140257793613248] AH00094: Command line: 'httpd -D FOREGROUND'
```

The container keeps running and the logs are displayed in your terminal.

In a web browser, go to `http://localhost:8080`. Verify that the HTTP server is running at the 8080 port.

Go back to your command-line terminal. Press **Ctrl**+**c** to stop the container.

```
...output omitted...
^C
[Thu Apr 21 13:00:52.516503 2022] [mpm_event:notice] [pid 1:tid 140481583287744] AH00491: caught SIGTERM, shutting down
```

Create the container again by adding the `-d` option. The container runs in the background.

```
[student@workstation ~]$ podman run --rm -d \
  -p 8080:8080 registry.ocp4.example.com:8443/ubi9/httpd-24
c1db...d61a
```

Verify that the container is running in the background.

```
[student@workstation ~]$ podman ps
... registry.ocp4.example.com:8443/ubi9/httpd-24:latest ... 0.0.0.0:8080->8080/tcp, ...
```

4. Use Podman Desktop to list images and containers.

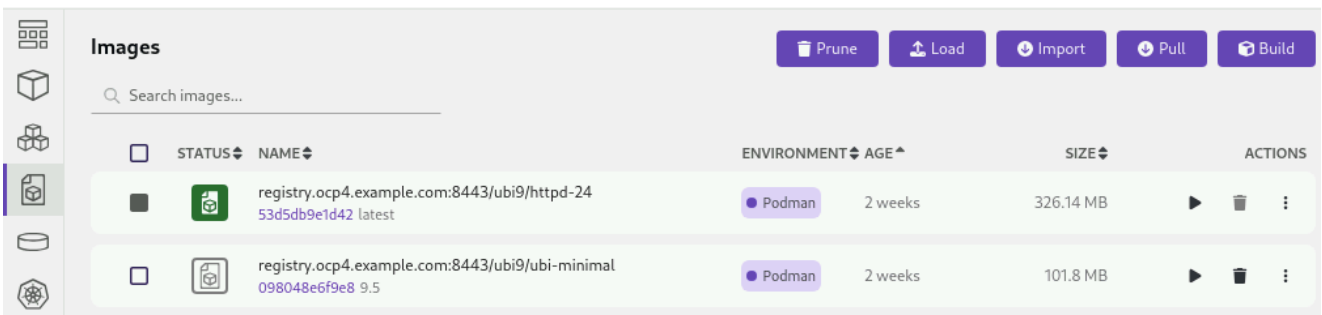
   Open Podman Desktop.

   ```
   [student@workstation ~]$ podman-desktop
   ```

   If you receive the Welcome Page, click **Skip** to go the full user interface.

   Click **Containers** in the Podman Desktop navigation panel. The list includes the containers that you created in the exercise. Verify that the Apache HTTP container is running.

   Click **Images**. The list of local images displays the `ubi9/ubi-minimal` and `ubi9/httpd-24` images.

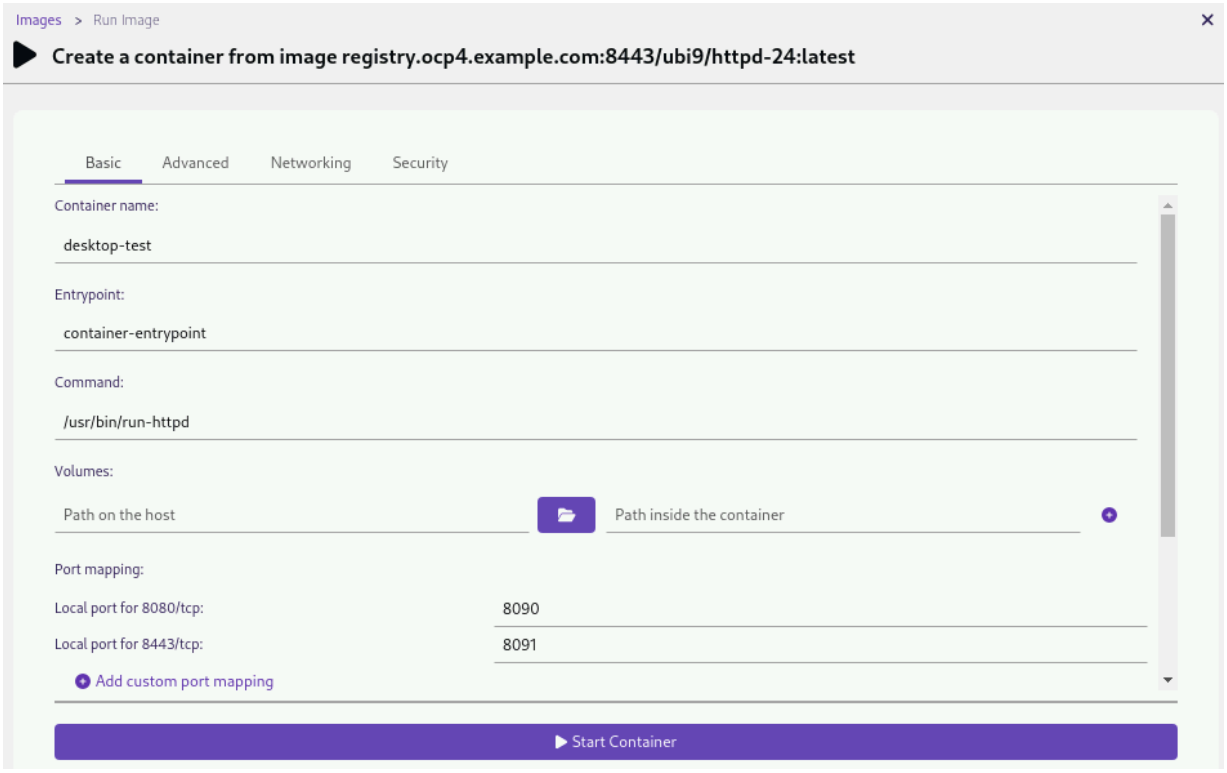


5. Use Podman Desktop to start another Apache HTTP container that maps its port to port 8090 in the workstation.

   Click the ▸ icon of the `ubi9/httpd-24` image to run a new container based on this image.

   In the container creation form, enter the following values:

   - Container name: `desktop-test`
   - Local port for 8080/tcp: `8090`
   - Local port for 8443/tcp: `8091`

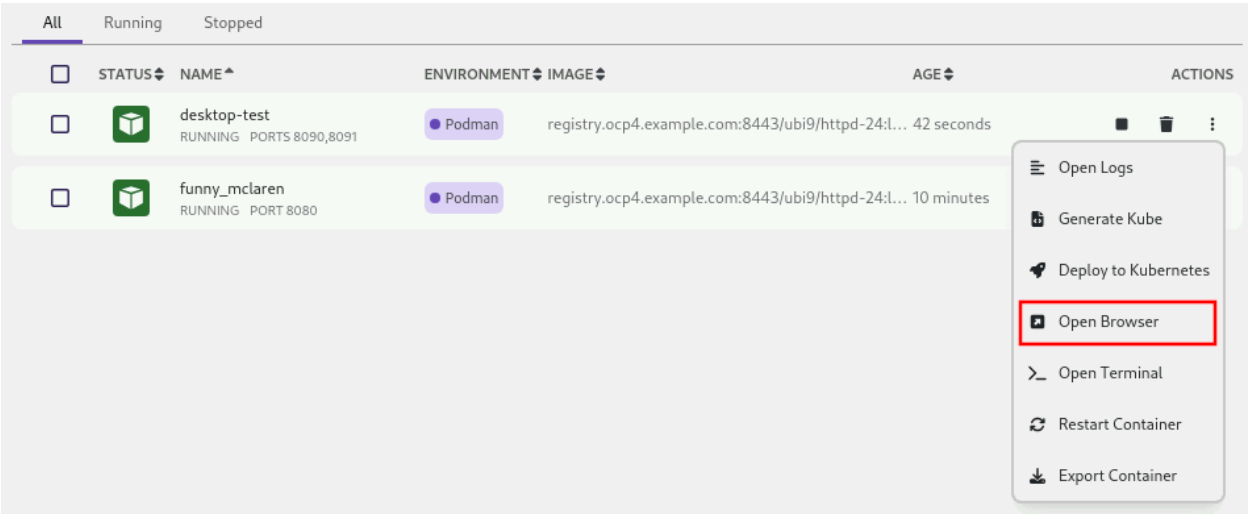


   Click **Start Container**.

> **NOTE**
>
> You can safely ignore the SELinux security warning that displays at the top of the desktop.

   In the containers list, verify that `desktop-test` is running.

   In the `desktop-test` container, scroll to the right if necessary, click ⋮ → **Open Browser**, and then click **Yes** in the confirmation dialog box.



   Verify that the browser can access the HTTP server at `http://localhost:8090`.

Return to Podman Desktop. In the `desktop-test` container, click the **Delete Container** icon to delete the container, and then close Podman Desktop.

Finish

On the `workstation` machine, use the `lab` command to complete this exercise. This is important to ensure that resources from previous exercises do not impact upcoming exercises.

```
[student@workstation ~]$ lab finish basics-creating
```