Guided Exercise: Managing the Container Lifecycle

Manage the lifecycle of a container that runs an Apache HTTP server.

Outcomes

You should be able to:

- Get detailed information about a container.
- Stop containers.
- Restart a stopped container.
- Delete containers.

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

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

Instructions

1. Create a container that runs an Apache HTTP server in the background.

   Execute the `podman run` command to create the container. Expose the `8080` port and use the `registry.ocp4.example.com:8443/ubi8/httpd-24` image.

   ```
   [student@workstation ~]$ podman run --name httpd -d -p \
     8080:8080 registry.ocp4.example.com:8443/ubi9/httpd-24
   Trying to pull registry.ocp4.example.com:8443/ubi9/httpd-24:latest...
   ...output omitted...
   ```

2. Verify that the container is running.

   Use `podman ps` to list all the running containers.

   ```
   [student@workstation ~]$ podman ps
   CONTAINER ID  IMAGE        COMMAND         CREATED STATUS PORTS           NAMES
   ID            registry...  ...run-http...  ...ago  Up...  0.0.0.0:8080... httpd
   ```

   Use the `podman inspect` command to get the `Status` field, which indicates whether the container is running.

   ```
   [student@workstation ~]$ podman inspect --format='{{.State.Status}}' httpd
   running
   ```

   Verify that the container is running by using the `Running` field.

   ```
   [student@workstation ~]$ podman inspect --format='{{.State.Running}}' httpd
   true
   ```

   In a web browser, go to `http://localhost:8080` to verify that the Apache server is running.

3. Stop the container.

   Return to your command-line terminal. Use the container name to stop the container.

   ```
   [student@workstation ~]$ podman stop httpd
   httpd
   ```

   Retrieve the `Status` of the container.

   ```
   [student@workstation ~]$ podman inspect --format='{{.State.Status}}' httpd
   exited
   ```

   Verify that the container is not running.

   ```
   [student@workstation ~]$ podman inspect --format='{{.State.Running}}' httpd
   false
   ```

   In a web browser, verify that the Apache server is no longer at the 8080 port.

4. Restart the container.

   In your command-line terminal, use the `podman restart` command to restart the container.

   ```
   [student@workstation ~]$ podman restart httpd
   httpd
   ```

   Verify that the container is running again.

   ```
   [student@workstation ~]$ podman ps
   CONTAINER ID  IMAGE        COMMAND         CREATED STATUS PORTS       NAMES
   ID            registry...  ...run-http...  ...ago  Up...  0.0.0.0:8080... httpd
   ```

   In a web browser, verify that the Apache server is running by navigating to `http://localhost:8080`.

5. Remove the container.

   Try to remove the running container by using the `podman rm` command.

   ```
   [student@workstation ~]$ podman rm httpd
   Error: cannot remove container CONTAINER_ID as it is running - running or paused containers cannot be removed without forc
   e: container state improper
   ```

   Podman requires that you to stop the container before you can remove it. However, you can force the removal of the container by adding the `--force` flag.

   Force the removal of the container.

   ```
   [student@workstation ~]$ podman rm httpd --force
   httpd
   ```

6. Verify that the container has been removed.

   Try to retrieve the status of the container.

   ```
   [student@workstation ~]$ podman inspect --format='{{.State.Running}}' httpd
   Error: no such object: "httpd"
   ```

   Because you removed the container, you can no longer retrieve its status.

7. Forcefully stop a container that does not respond to the `SIGTERM` signal.

   Start a new container in the background by using the `registry.ocp4.example.com:8443/redhattraining/podman-greeter-ignore-sigterm` image. The application ignores `SIGTERM` signals.

   ```
   [student@workstation ~]$ podman run --name greeter -d \
     registry.ocp4.example.com:8443/redhattraining/podman-greeter-ignore-sigterm
   ...output omitted...
   f919...e69b
   ```

   Try to stop the container with a grace period of `5` seconds.

   ```
   [student@workstation ~]$ podman stop greeter --time=5
   WARN[0005] StopSignal SIGTERM failed to stop container greeter in 5 seconds, resorting to SIGKILL
   greeter
   ```

   Podman sends a `SIGTERM` signal to stop the container gracefully, but the application ignores the signal. After 5 seconds, Podman sends a `SIGKILL` signal to the container. The `--time` flag indicates the time that Podman waits before sending a `SIGKILL` signal to forcefully stop the container.

   Restart the container.

   ```
   [student@workstation ~]$ podman restart greeter
   greeter
   ```

   Use the `podman kill` command to directly send a `SIGKILL` signal and stop the container forcefully.

   ```
   [student@workstation ~]$ podman kill greeter
   greeter
   ```

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-lifecycle
```