Lab

Creating Containerized Services

Performance Checklist

In this lab, you create an Apache HTTP Server container with a custom welcome page.

Outcomes

You should be able to start and customize a container using a container image.

Before You Begin

Open a terminal on **workstation** as the **student** user and run the following command:

```
[student@workstation ~]$ lab container-review start
```

1. Start a container named **httpd-basic** in the background, and forward port 8080 to port 80 in the container. Use the **redhattraining/httpd-parent** container image with the **2.4** tag.

   Note

   Use the **-p 8080:80** option with **sudo podman run** command to forward the port.

   This command starts the Apache HTTP server in the background and returns to the Bash prompt.

2. Test the **httpd-basic** container.

   From **workstation**, attempt to access `http://localhost:8080` using any web browser.

   An **Hello from the httpd-parent container!** message is displayed, which is the **index.html** page from the Apache HTTP server container running on **workstation**.

3. Customize the **httpd-basic** container to display **Hello World** as the message. The container's message is stored in the file **/var/www/html/index.html**.

   3.1. Start a Bash session inside the container.

   3.2. From the Bash session, verify the **index.html** file under **/var/www/html** directory using the **ls -la** command.

   3.3. Change the **index.html** file to contain the text **Hello World**, replacing all of the existing content.

   3.4. Attempt to access `http://localhost:8080` again, and verify that the web page has been updated.

Evaluation

Grade your work by running the **`lab container-review grade`** command on your **`workstation`** machine. Correct any reported failures and rerun the script until successful.

```
[student@workstation ~]$ lab container-review grade
```

Finish

On **`workstation`**, run the **`lab container-review finish`** script to complete this lab.

```
[student@workstation ~]$ lab container-review finish
```

This concludes the lab.

Solution

Creating Containerized Services

Performance Checklist

In this lab, you create an Apache HTTP Server container with a custom welcome page.

Outcomes

You should be able to start and customize a container using a container image.

Before You Begin

Open a terminal on **workstation** as the **student** user and run the following command:

```
[student@workstation ~]$ lab container-review start
```

1. Start a container named **httpd-basic** in the background, and forward port 8080 to port 80 in the container. Use the **redhattraining/httpd-parent** container image with the **2.4** tag.

Note

Use the **-p 8080:80** option with **sudo podman run** command to forward the port.

Run the following command:

```
[student@workstation ~]$ sudo podman run -d -p 8080:80 \
> --name httpd-basic redhattraining/httpd-parent:2.4
...output omitted...
Copying blob sha256:743f2d6...output omitted...
 21.45 MB / 21.45 MB [====================================================] 1s
Copying blob sha256:c92eb69...output omitted...
 155 B / 155 B [==========================================================] 0s
Copying blob sha256:2211b05...output omitted...
 9.86 MB / 9.86 MB [======================================================] 0s
Copying blob sha256:aed1801...output omitted...
 15.78 MB / 15.78 MB [====================================================] 1s
Copying blob sha256:7c472a4...output omitted...
 300 B / 300 B [==========================================================] 0s
Copying config sha256:b7cc370...output omitted...
 7.18 KB / 7.18 KB [======================================================] 0s
Writing manifest to image destination
Storing signatures
b51444e3b1d7aaf94b3a4a54485d76a0a094cbfac89c287d360890a3d2779a5a
```

This command starts the Apache HTTP server in the background and returns to the Bash prompt.

2. Test the **`httpd-basic`** container.

   From **`workstation`**, attempt to access `http://localhost:8080` using any web browser.

   An **`Hello from the httpd-parent container!`** message is displayed, which is the **`index.html`** page from the Apache HTTP server container running on **`workstation`**.

```
[student@workstation ~]$ curl http://localhost:8080
Hello from the httpd-parent container!
```

3. Customize the **`httpd-basic`** container to display **`Hello World`** as the message. The container's message is stored in the file **`/var/www/html/index.html`**.

   3.1. Start a Bash session inside the container.

   Run the following command:

```
[student@workstation ~]$ sudo podman exec -it httpd-basic /bin/bash
bash-4.4#
```

   3.2. From the Bash session, verify the **`index.html`** file under **`/var/www/html`** directory using the **`ls -la`** command.

```
bash-4.4# ls -la /var/www/html
total 4
drwxr-xr-x. 2 root root 24 Jun 12 11:58 .
drwxr-xr-x. 4 root root 33 Jun 12 11:58 ..
-rw-r--r--. 1 root root 39 Jun 12 11:58 index.html
```

   3.3. Change the **`index.html`** file to contain the text **`Hello World`**, replacing all of the existing content.

   From the Bash session in the container, run the following command:

```
bash-4.4# echo "Hello World" > /var/www/html/index.html
```

   3.4. Attempt to access `http://localhost:8080` again, and verify that the web page has been updated.

```
bash-4.4# exit
[student@workstation ~]$ curl http://localhost:8080
Hello World
```

Evaluation

Grade your work by running the **`lab container-review grade`** command on your **`workstation`** machine. Correct any reported failures and rerun the script until successful.

```
[student@workstation ~]$ lab container-review grade
```

Finish

On **`workstation`**, run the **`lab container-review finish`** script to complete this lab.

```
[student@workstation ~]$ lab container-review finish
```

This concludes the lab.