Docker Hands-On Lab 10 — Building a Custom Apache Image

Trainer: Sandeep Kumar Sharma

Duration: 60–90 minutes

Goal: In this lab, you'll learn to build a **custom Docker image** step-by-step using a Dockerfile. You'll understand every instruction, the build process, and how to test the resulting container. This example uses Ubuntu as the base and installs Apache web server with a custom web page.

---

1) Understanding the Scenario

We will create a custom image named **sandeep/apache:v1** using the following Dockerfile:

```
FROM ubuntu:18.04
MAINTAINER sandeep
RUN apt-get update -y
RUN apt-get install apache2 -y
RUN apt-get install apache2-utils -y
RUN echo "This is Web application Running inside Container" > /var/www/html/
index.html
EXPOSE 80
CMD ["apache2ctl", "-D", "FOREGROUND"]
```

---

2) Explanation of Each Instruction

Instruction	Description
FROM ubuntu:18.04	This defines the base image. All subsequent instructions build on top of this image.
MAINTAINER sandeep	Provides metadata about the image author. (Note: `LABEL maintainer` is preferred in newer Docker versions.)
RUN apt-get update -y	Updates the package list to ensure the latest versions are available.
RUN apt-get install apache2 -y	Installs the Apache web server inside the image.
RUN apt-get install apache2-utils -y	Installs Apache utilities for admin tasks like benchmarking or authentication.

Instruction	Description
RUN echo ... > /var/www/html/index.html	Creates a simple web page that will be served by Apache when accessed.
EXPOSE 80	Informs Docker that this container listens on port 80 (the default HTTP port).
CMD ["apache2ctl", "-D", "FOREGROUND"]	Tells Docker to start Apache in the foreground (important so the container keeps running).

---

3) Step-by-Step Hands-On Lab

Step 1: Prepare the Working Directory

```
mkdir ~/docker-apache-lab
cd ~/docker-apache-lab
```

Step 2: Create the Dockerfile

Create a file named `Dockerfile` and paste the contents from above.

```
nano Dockerfile
```

(Paste the content, save, and exit.)

Step 3: Build the Image

```
docker build -t sandeep/apache:v1 .
```

Docker will: 1. Pull the base image `ubuntu:18.04`. 2. Install Apache. 3. Create a custom web page. 4. Expose port 80.

Expected output: Each `RUN` instruction creates a new layer.

Step 4: Verify the Image

```
docker images
```

You should see:

```
REPOSITORY       TAG       IMAGE ID       CREATED          SIZE
sandeep/apache   v1        <id>           a few seconds ago   <size>
```

Step 5: Run a Container from the Image

```
docker run -d --name sandeep-web -p 8080:80 sandeep/apache:v1
```

This command: - Runs the container in detached mode (`-d`). - Names the container `sandeep-web`. - Maps host port 8080 to container port 80.

Step 6: Test in Browser

Visit `http://localhost:8080` (or your VM IP if remote). You should see:

```
This is Web application Running inside Container
```

---

4) Inspect and Verify

View Logs

```
docker logs sandeep-web
```

Inspect Container Details

```
docker inspect sandeep-web
```

Access the Container Shell

```
docker exec -it sandeep-web /bin/bash
```

Once inside, verify Apache service:

```
service apache2 status
```

---

5) Modify and Rebuild the Image

Suppose you want to change the webpage text. Edit the Dockerfile line:

```
RUN echo "Welcome to Sandeep's Custom Web Server" > /var/www/html/index.html
```

Rebuild:

```
docker build -t sandeep/apache:v2 .
```

Run again:

```
docker run -d --name sandeep-web2 -p 9090:80 sandeep/apache:v2
```

Visit `http://localhost:9090` to see the updated message.

---

6) Explore Layers and Optimization

Check image layers:

```
docker history sandeep/apache:v1
```

Each `RUN` creates a layer. To reduce size and improve performance, you can merge commands:

```
RUN apt-get update && apt-get install -y apache2 apache2-utils &&
    echo "This is Web application Running inside Container" > /var/www/html/
index.html
```

This reduces layers and build time.

Rebuild:

```
docker build -t sandeep/apache:optimized .
```

---

7) Cleanup

```
docker stop sandeep-web sandeep-web2
```

```
docker rm sandeep-web sandeep-web2
```

```
docker rmi sandeep/apache:v1 sandeep/apache:v2 sandeep/apache:optimized
```

---

8) Practice Tasks

1. Modify the base image from Ubuntu to `debian:latest` and rebuild.
2. Change the port exposure from 80 to 8081 and verify access.
3. Add a `LABEL` instruction with version and author details.
4. Create an HTML page with more content (like a welcome header and paragraph).
5. Push your image to Docker Hub with the tag `sandeep/apache:practice`.

---

9) Summary

- You learned to create a Docker image from scratch using a Dockerfile.
- Each instruction adds a new image layer.
- `RUN`, `EXPOSE`, and `CMD` define how the image is built and executed.
- You built and tested a working Apache web server image.
- Optimization helps reduce image size and build time.

✅**Checkpoint:** You successfully built and customized your own Apache image and understood every part of the Dockerfile!