Lab #1 – Docker Getting Started

Overview

Application containerization is an OS-level virtualization method used to deploy and run applications. Multiple isolated applications or services run on a single host and access the same OS kernel. Containers work on bare-metal systems, cloud instances and virtual machines, across Linux and select Windows and Mac OSes.

Docker is a platform that helps implement application containerization. In this lab, you will develop an application and containerize it by building a Docker image. You will answer questions about pushing the image and Docker architecture.

Preparation

- Install Node on Ubuntu; this will be used to develop your application. For more information on how to install Node on Ubuntu: https://www.geeksforgeeks.org/installation-of-node-js-on-linux/
- Install Docker on Ubuntu using the default repositories; this will be used to prepare your docker image and run your application within a container. For more information on how to install Docker on Ubuntu: https://howtoinstall.co/en/docker.io

Part 1

For part 1, you will create a Node application in the form of a single JavaScript file that outputs to Console. Use the following information:

Filename: `app.js`
Code: `console.log("Hello Docker!");`

Check your Node installation with the command "`node -v`" – it should print a version number

Check your application with the command "`node app.js`" – it should output "Hello Docker!"

Once complete, adjust the code to output something other than "Hello Docker!". The way to test your changes is the same and should reflect in the output on screen. Include the command and output in your screenshot to prove your application is working.

Screenshots

Show a screenshot or screenshots that are identifiable as yourself that show the work was completed.

Reflection

Write a reflection with this part of the lab. What are the components of this application? What would you need to do to containerize this application? What kind of benefits do you foresee if it were to be containerized? Use some or all of these questions to inspire your reflection on the work done.

Part 2

For part 2, you will containerize your application by creating a Docker configuration file, building a Docker image and running it within a container. Use the following information:

Docker Configuration Filename: `Dockerfile`
Dockerfile Contents:

```
FROM node:alpine
COPY . /app
CMD node /app/app.js
```

Check your Docker installation with the command "`docker version`" – it should print version information

Build your Docker image with the command "`docker build -t hello-docker .`" from within your application folder – the process should complete successfully

Once the image is built, use the command "`docker images`" to list the images on your host.

Run your containerized application with the command "`docker run hello-docker`" – the application should run successfully

Screenshots

Show a screenshot or screenshots that are identifiable as yourself that show the work was completed.

Reflection

Write a reflection with this part of the lab. How does the containerized application differ from when it wasn't containerized? Can you explain the steps in the Docker configuration file? When listing images what information is being displayed and what do they mean? How do these steps help in building, running and shipping your application? Use some or all of these questions to inspire your reflection on the work done.

Part 3

For part 3, you will publish your image to the public registry provided by Docker, called Docker Hub. Docker Hub is a service provided by Docker for finding and sharing container images. It is the world's largest repository of container images. Use the following steps to complete this part:

1. **Sign up for a Docker account: You have to create a Docker ID; this grants you access to Docker Hub repositories. You will need a Docker ID to share images on Docker Hub**

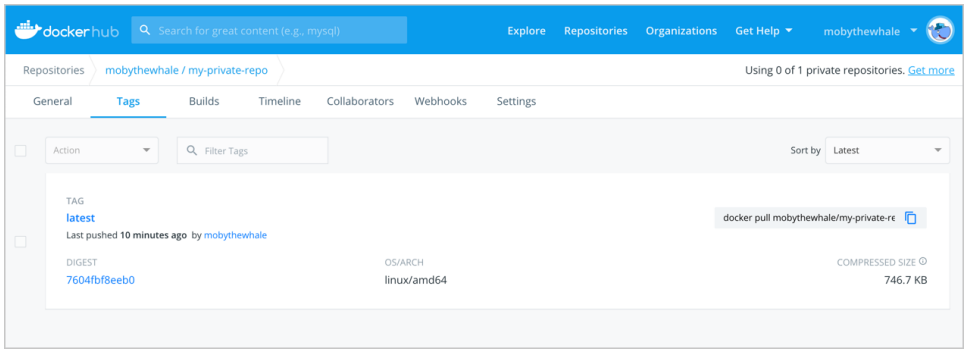
2. **Build and push a container image to Docker Hub from your computer**

    a. Run "`docker build -t <your_username>/hello-docker .`" to build your Docker image

    b. Run "`docker run <your_username>/hello-docker`" to test your Docker image locally

    c. Run "`docker push <your_username>/hello-docker`" to push your Docker image to Docker Hub

Your repository in Docker Hub should now display a new latest tag under Tags:



Screenshots

Show a screenshot or screenshots that are identifiable as yourself that show the work was completed.

Reflection

Write a reflection with this part of the lab. What is the benefit of publishing your image to Docker Hub? What other services does Docker Hub provide? What can one do if they want to use a private registry instead? What part does the registry play in "building, running and shipping" your application? Use some or all of these questions to inspire your reflection on the work done.