HW6: Programming Assignment v03/31/2022 8:32Am

WORKING WITH DOCKER CONTAINERS AND MICROSERVICES

In this assignment you will create a server and multiple clients, that will be deployed in a docker container. Then you will automate their deployment. Completing this assignment will give you a feel for how many companies can now run modern containerized micro web services.

Due Date: April 14,2022

1. Description of Task

For this assignment you will be working to build and automate the deployment of two programs (microservices) which will require the following components. For this assignment you will deploy Docker on your personal computer. Before you get started you will need to download Docker here: https://docs.docker.com/get-docker/ Docker works with Windows, Mac, and Linux.

The server will print the names received from 3 different clients.

`Server:` Basic python server. [10]

1. Minimal amount of code (~10 lines)
2. Listens on port 5000
3. The server receives names from different clients.
4. Prints out the names received from the clients.
5. This file should be inside a folder called **app.**

Note: Flask would be a better and easier option for communication with the server. You can start here for information on flask: https://flask.palletsprojects.com/en/2.0.x/quickstart/

`Docker Container for Server:` Docker container that contains the python server. [10]

1. Create a Dockerfile (config file that specifies how container will be built) (~6 lines of text)
2. Create a requirements.txt file (specifies python dependencies for pip to be installed in the server)
3. Put both files and the app folder with the python server code in a folder called server.

`Client:` Basic python Client. [10]

1. Minimal amount of code (~9 lines of text)
2. Does a POST request on http://localhost:5000 to send the name to server
3. Prints the name sent to the server
4. This file should be inside a folder called app.

`Docker Container for Client:` Docker container that contains the python server. [10]

1. Create a Dockerfile (script that specifies how container will be built) (~6 lines of text)
2. Create a requirements.txt file that specifies the python dependencies for client (This is not necessary but depends on your implementation).
3. Put both files and the app folder with the python client code in a folder called client.

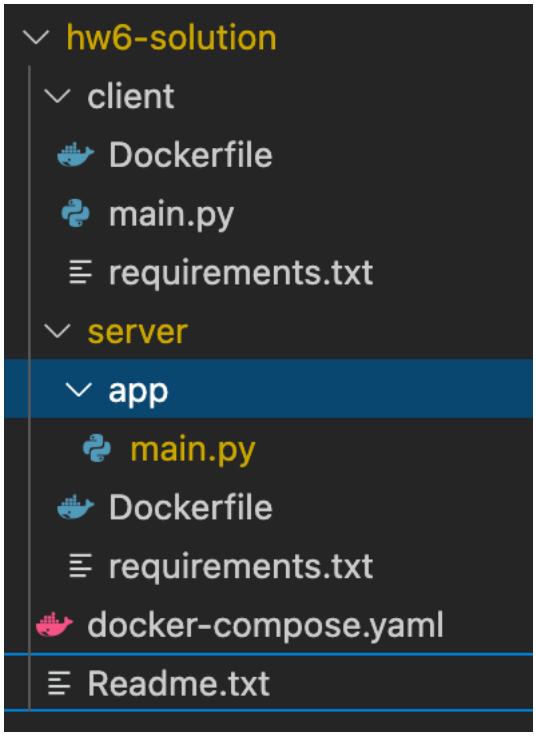
`Docker Compose:` Specifications for deploying the clients and server containers. [10]

1. Create docker-compose.yml
2. Minimal amount of code (~11 lines of text)
3. Create three different clients sending the names required as arguments.
4. Create a folder called hw6 that contains the docker-compose.yml as well as the server and client folders created earlier.
5. When run with: 'docker-compose up' both clients and server containers should be built and run which will cause the server to print the messages that it receives from the multiple clients.

README.txt [40]

Do not forget to implement README.txt parallel to your docker-compose.yaml, the HW6 folder structure should be like the diagram below.

HW6 folder structure:



Hints:

1. Answer the questions in section 4 first to get a feeling for how docker works. You can find the answers by looking through dockers website https://docs.docker.com/, typing 'docker' into the command line to look at the built-in help docs, or googling the question.
2. This assignment will be mostly about getting comfortable with the different components of the docker ecosystem, the actual code will be minimal and many of the components can be found already completed in Dockers documentation or on different internet sites. Putting everything together is what will take a bit of work.
3. Starting points for the code in this program can be found in several places on dockers websites as well as the internet in general. Here are some search terms you might want to try besides the obvious, ex. dockerfile, docker-compose, requirements.txt, etc.
    a. docker python server
    b. pythons get request
    c. python print response text
    d. docker-compose network host
4. Use Google / search for any error messages to address them.

2. Task Requirements

1. Answer the questions in Part 4 (40% of your grade)
2. Complete the components described in Part 1 (50% of your grade)
3. When the command docker-compose up is run output like the following should be displayed. Because this project has some flexibility the output does not have to be identical but the following items in the output should be visible: output related to the server container being built, output related to the client being built, output related to the client sending names to the server, outputs related to the names printed by the server (highlighted red). (10% of your grade)

3. Files Provided

This writeup is the only file provided with this assignment.

4. Questions: Include the answers to the following short answer questions in a file called README.txt that you will create and included in your hw6 folder.

1. What is a docker container? [2]
2. What is the difference between a container and a virtual machine? [2]
3. What is the purpose of a Dockerfile? [2]
4. What is the purpose of a requirements.txt file? [2]
5. What is the purpose of a docker-compose.yml file? [2]
6. What is the difference between a docker image and docker container? [2]
7. What command can be used to create an image from a Dockerfile? [2]
8. What command will start a docker container? [2]
9. What command will stop a docker container? [2]
10. What command will remove a docker container? Image? [2]
11. What command will list all running docker containers? all containers? [3]
12. What command will list all docker images? [2]
13. What command do you use to deploy docker containers using information in the docker-compose.yml file? [5]
14. How can you specify in the docker-compose.yml file that you want docker containers to use the hosts network? [5]
15. How can you specify in the docker-compose.yml file where the Dockerfile for a particular container is found? [5]

5. Example Outputs (Note: - The names of servers and clients may be different, the ids of containers can be different on your system, However the IMPORTANT line required for output are highlighted.)

C:\Users\\user\hw6$ docker-compose up --build

[hw6-solution_client3 internal] load build definition from Dockerfile
=> => transferring dockerfile: 127B
=> [hw6-solution_server internal] load build definition from Dockerfile
=> => transferring dockerfile: 127B
=> [hw6-solution_client2 internal] load build definition from Dockerfile
=> => transferring dockerfile: 127B
=> [hw6-solution_client1 internal] load build definition from Dockerfile
=> => transferring dockerfile: 127B

Container hw6-solution-client3-1 Created 0.0s
⠿ Container hw6-solution-client2-1 Created 0.0s
⠿ Container hw6-solution-server-1 Created 0.0s
⠿ Container hw6-solution-client1-1 Created 0.0s
Attaching to hw6-solution-client1-1, hw6-solution-client2-1, hw6-solution-client3-1, hw6-solution-server-1

hw6-solution-client1-1 | my name is Alice
hw6-solution-client2-1 | my name is Bob
hw6-solution-client3-1 | my name is Charlie
hw6-solution-server-1 | Client Name is Alice
hw6-solution-server-1 | Client Name is Bob
hw6-solution-client1-1 | Request Sent!
hw6-solution-client2-1 | Request Sent!
hw6-solution-server-1 | Client Name is Charlie
hw6-solution-client3-1 | Request Sent!
hw6-solution-client2-1 exited with code 0
hw6-solution-client1-1 exited with code 0
hw6-solution-client3-1 exited with code 0

hw6-solution % docker compose down
[+] Running 5/5
⠿ Container hw6-solution-client3-1 Removed 0.1s
⠿ Container hw6-solution-server-1 Removed 0.1s
⠿ Container hw6-solution-client1-1 Removed 0.1s
⠿ Container hw6-solution-client2-1 Removed 0.1s
⠿ Network hw6-solution_default Removed

6. What to Submit

Use the CS370 Canvas to submit a single .zip or .tar file that contains:

- The folder hw6 with the contents and structure specified in part 1

For this and all other assignments, ensure that you have submitted a valid .zip/.tar file. After submitting your file, you can download it and examine to make sure it is indeed a valid zip/tar file, by trying to extract it.

Filename Convention: The archive file must be named as: <FirstName>-<LastName>-HW6. <tar/zip>. E.g., if you are John Doe and submitting for assignment 1, then the tar file should be named John-Doe-HW1.tar

7. Grading

For this assignment, your code should run on your computer (or any computer) that is running the latest version on Docker.

IMPORTANT: For the output (Task 2.3) we care most about the output from the client and server which is in yellow/red above.

The grading will be done on a 100-point scale. The points are broken up as follows:

Objective	Points
Task 2.1 (answering Questions from Part 4)	40 points
Task 2.2 (completing the files specified in Part 1)	50 points
Task 2.3 (running docker-compose up from your hw6 folder creates correct output)	10 points

8. Late Policy

Click here for the class policy on submitting late assignments.

Revisions: Any revisions in the assignment will be noted below.