Docker Exercise

General

In this exercise you are required to create a mini topology of 2 Docker containers.

Criteria for Success

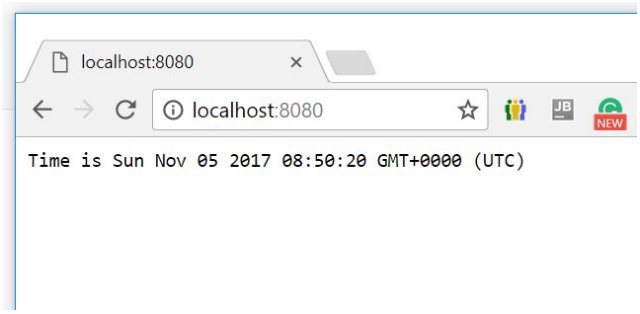
These are the criteria for success, by importance:

(1) It works.
(2) Code is clean.
(3) Improvement suggestions make sense

Tasks

(1) Create "Timeservice" container

The "Timerservice" is a docker image that contains a web server which returns a simple screen with the current time:



All source files for your container (e.g. dockerfile, scripts, etc.) should be put in folder `~/exam/timerservice`.

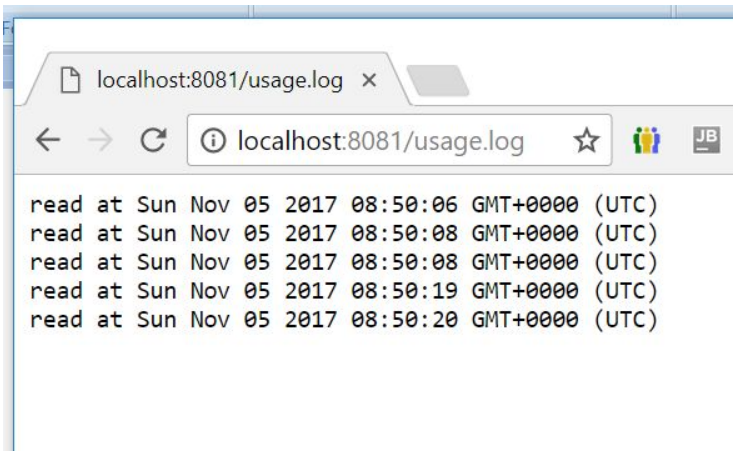
There are no limitations on the technology you use inside the docker container: python, node, ruby or whatever – use what makes you efficient.

(2) Create "Loggerservice" container

We would like to monitor calls to the "Timerservice" using a new container called "loggerservice".

- Modify "timerservice" so that it logs each time a user accesses it to the "loggerservice" (Note: this is not the web servers' access.log)

- The “loggerservice” is also a container with a web server. It displays the content of the “timerservice” usage log at url `/usage.log`:



All source files for your container (e.g. dockerfile, scripts, etc.) should be put in folder `~/exam/loggerservice`.

There are no limitations on the technology you use inside the docker container: python, node, ruby or whatever – use what makes you efficient.

(3) Create Docker-Compose Topology

Create a docker-compose file which will start both servers mentioned and connect them:

- Timerservice on port 8080
- Loggerservice on port 8081
- Loggerservice displays the logs of Timerservice

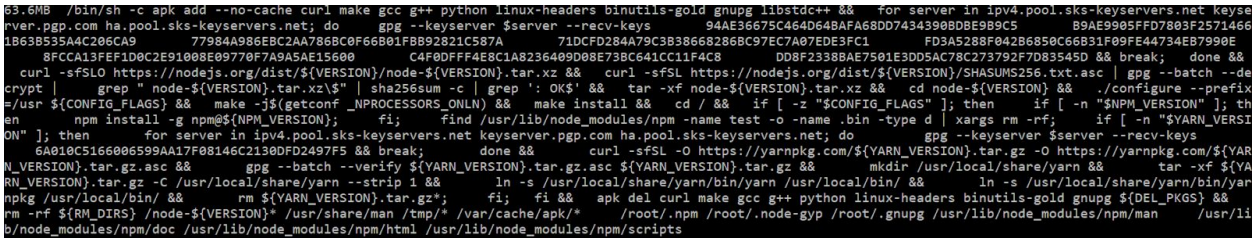
The topology should be started using single “`docker-compose up`” command in `~/exam`

(4) Scripts

(a) Write a script named “time-layers” that lists all the layers in your **timerservice** docker image, ordered by their size:

```
0B      /bin/sh -c #(nop)  CMD ["/bin/sh"]
0B      /bin/sh -c #(nop)  CMD ["node" "/src/server.…
0B      /bin/sh -c #(nop)  ENV VERSION=v10.7.0 NPM_V…
0B      /bin/sh -c #(nop)  EXPOSE 4000
486B    /bin/sh -c cd /src && npm install
826B    /bin/sh -c #(nop) COPY dir:ad8b10e40ec0ca2ed…
4.2MB   /bin/sh -c #(nop) ADD file:6ee19b92d5cb1bf14…
63.6MB  /bin/sh -c apk add --no-cache curl make gcc …
```

(b) What is the most expensive layer? in our example it was the addition of packages related to nodejs:



Make sure it is indeed last in time-layers execution

Submission

Make sure all sources are in the ~/exam folder

Once you are done, hand your machine for inspection.

Tester will execute

(1) `docker-compose up -d` from `~/exam`

(2) `time-layers` from `~/exam`

Make sure it works before submitting!