Lab 2 - Setup Web Server in Docker

Setting up web server is one of the common use case in Docker. Remember that we are using **docker-machine** library as docker can only support Linux kernel natively. Therefore, we are running a Linux VM in Windows, the Docker host address is hence the address of the Linux VM.

Start an Nginx container

Let's start a Nginx container

```
$ docker run -d -P --name web nginx
```

The `-d` prompt indicates that it is run as a detached mode, and it will keep running in the background (even after the run command completes). The `-P` flag exposed ports from container to our local host, so that it is accessible from our computer.

> p/s: You may use the `docker ps` command to ensure the instance is up and running.

To view the container's ports that are exposed, we can use the following command.

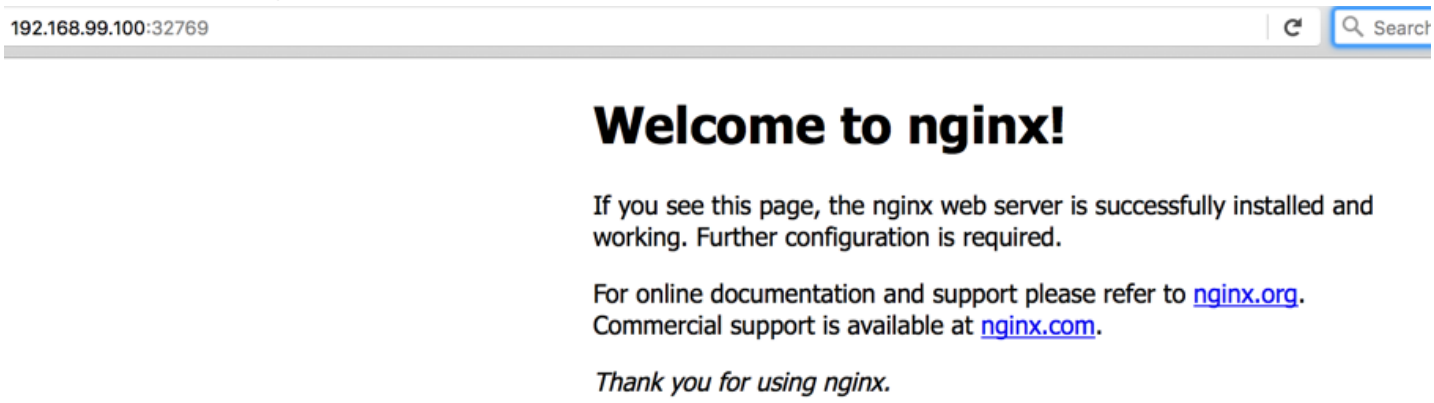
```
$ docker port web
```

and we should get the following results.

```
443/tcp -> 0.0.0.0:32768
80/tcp -> 0.0.0.0:32769
```

This indicates that the port 32678 and 32769 is exposed. So, supposedly, we should be able to browse our website using http://localhost:32769. Note that the port number might vary for individual system, please change it to the port number as shown on your screen.

Now you are expecting to see a welcome screen like below:



However, you see an error screen. What went wrong here? The reason is because if you are running Docker on a non-Linux machine (e.g. Windows, OS X), you are actually running on a virtual machine, therefore, we will need to use the actual IP address of the VM, instead of localhost. To find out the address of the VM, use the following command.

```
$ docker-machine ip default
```

Enter the IP address and the port into the address bar of your browser, you should see the screen showing that the Nginx is up and running.

Let's stop remove and the container, and prepare for the next task.

```
$ docker stop web
$ docker rm web
```

Create a container with a link to local folder

In this activity, what we want is to start a container, but with a folder that links to our host folder. First, switch to a folder.

Next, we want to create a directory name `site`. Create a file `index.html`, with the following content.

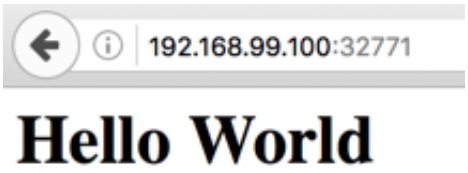
```
<h1>Welcome to Docker</h1>
```

Start a new Nginx container.

```
docker run -d -P -v $HOME/site:/usr/share/nginx/html --name mysite nginx
```

Enter the url in your browser (double check and make sure the address and port is correct)

You should see a screen similar to below.



Hello World

Task: You can add another file to the `site` folder, and your website content will be updated in real time.

> Note: It is a known bug that if you modified the file in real time, it will not reflect the changes, as the content are being cached. However, adding file will not be affected.

Lab Exercise

Remove the container, and start another one with your own content. Also try to include other static file types (e.g. jpg, css and js)