Docker – Tomcat as Container

Objective

To run **"tomcat"** application as docker container.

1. To pull the docker image "tomcat" from the "hub.docker.com"

Currently no "tomcat" image is available.

```
[root@localhost ~]# docker image ls
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
docker.io/hello-world   latest              fce289e99eb9        7 months ago        1.84 kB
[root@localhost ~]#
```

Run the below command to pull it.

$ docker pull tomcat

Note:-- Make sure the VM has the internet access.

```
[root@localhost ~]# docker pull tomcat
Using default tag: latest
Trying to pull repository docker.io/library/tomcat ...
latest: Pulling from docker.io/library/tomcat
a4d8138d0f6b: Pull complete
dbdc36973392: Pull complete
f59d6d019dd5: Pull complete
aaef3e026258: Pull complete
5e86b04a4500: Pull complete
1a6643a2873a: Pull complete
2ad1e30fc17c: Pull complete
16f4e6ee0ca6: Pull complete
928f4d662d23: Pull complete
b8d24294d525: Pull complete
Digest: sha256:2785fac92d1bcd69d98f2461c6799390555a41fd50d3f847b544368d594c637b
Status: Downloaded newer image for docker.io/tomcat:latest
[root@localhost ~]# docker image ls
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
docker.io/tomcat        latest              238e6d7313e3        12 days ago         506 MB
docker.io/hello-world   latest              fce289e99eb9        7 months ago        1.84 kB
[root@localhost ~]#
```

Since the image is not available locally, its going to pull from the internet.

2. Create an container for this image

```
[root@localhost ~]# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
[root@localhost ~]# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
[root@localhost ~]#
```

$ docker ps → shows only the running containers.

$ docker ps -a → Shows all the containers including the stopped once.

Currently there are no containers as the above image.

Let's run the below command

$ docker run tomcat

```
1 192.168.3.105:22
31-Jul-2019 01:55:37.092 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploy
plication directory [/usr/local/tomcat/webapps/examples]
31-Jul-2019 01:55:37.805 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploy
b application directory [/usr/local/tomcat/webapps/examples] has finished in [713] ms
31-Jul-2019 01:55:37.805 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploy
plication directory [/usr/local/tomcat/webapps/host-manager]
31-Jul-2019 01:55:37.899 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploy
b application directory [/usr/local/tomcat/webapps/host-manager] has finished in [93] ms
31-Jul-2019 01:55:37.899 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploy
plication directory [/usr/local/tomcat/webapps/manager]
31-Jul-2019 01:55:37.969 INFO [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploy
b application directory [/usr/local/tomcat/webapps/manager] has finished in [69] ms
31-Jul-2019 01:55:37.977 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["http-nio-8
31-Jul-2019 01:55:38.007 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["ajp-nio-80
31-Jul-2019 01:55:38.016 INFO [main] org.apache.catalina.startup.Catalina.start Server startup in 1851 ms
```

The container has got created,

But, the above screen shows that we don't have the access to the linux prompt/session anymore, which is used up by the docker container, because the docker process is running in the foreground.

To run the docker in the background.

Let's stop this container.

$ ctrl+c

```
31-Jul-2019 01:57:35.971 INFO [Thread-5] org.apache.coyote.AbstractProtocol.destroy Destroying ProtocolHandler ["aj
p-nio-8009"]
[root@localhost ~]# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS
     NAMES
[root@localhost ~]# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS                      PORTS
               NAMES
1666ca078ca7        tomcat              "catalina.sh run"   2 minutes ago       Exited (130) 11 seconds ago
               objective_heyrovsky
[root@localhost ~]#
```

Let's remove this container before we proceed.

Note:-- KEEP THE DOCKER ENGINE CLEAN.

$ docker container rm <image id>

```
[root@localhost ~]# docker container rm 1666ca078ca7
1666ca078ca7
[root@localhost ~]#
```

Now, running the same docker run command in the background.

$ docker run -d tomcat

```
[root@localhost ~]# docker run -d tomcat
336388b8656516534535c9abdf16b4277ff385ad9c023220fc12708a501ade78
[root@localhost ~]#
```

```
[root@localhost ~]# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS
     NAMES
336388b86565        tomcat              "catalina.sh run"   55 seconds ago      Up 54 seconds       8080/tcp
     stupefied_wiles
[root@localhost ~]#
```

This shows that the container is running and its running on port 8080.

3. To check the ip address of the container created.

For this we need to login to the container itself.

$ docker exec -it <container id> /bin/bash

```
[root@localhost ~]# docker exec -it 336388b86565 /bin/bash
root@336388b86565:/usr/local/tomcat#
```

Kindly observe that the prompt has changed with **"root@<container id>:/usr/local/tomcat"**

Means this folder is set as default working directory in the **"tomcat"** image.

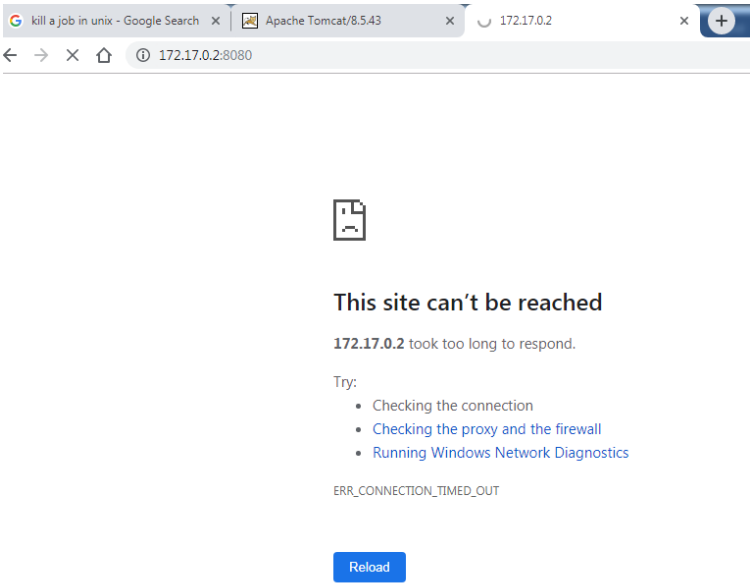
Run the below command to see the ip address.

$ ip add

Note: -- the regular "ifconfig" would not work as the docker images are built very light, by keeping all the extra services out of it.

```
root@336388b86565:/usr/local/tomcat# ip add
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group de
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
18: eth0@if19: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue stat
    link/ether 02:42:ac:11:00:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.17.0.2/16 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::42:acff:fe11:2/64 scope link
       valid_lft forever preferred_lft forever
root@336388b86565:/usr/local/tomcat#
```

Now to access this tomcat, when we try to use the above ip along with the port **"8080"(tomcat application port)**, it would not be reachable as the **tomcat app** is **inside** the container.



This shows that the tomcat application is not accessible.

With http://172.17.0.2:8080

Which is the ip address of the container itself.

To solve this problem, we need to map the container application ports to the base OS machine's Network port.

$ docker run -d -p 90:8080 tomcat

The port **"90"** is for the Host machine

Port **"8080"** is of the container

```
[root@localhost ~]# docker run -d -p 90:8080 tomcat
5f61c80b1666bd72f132a7cb0541977b68cd2308899a1145003e4cee2dee1d3b
[root@localhost ~]#
```

```
[root@localhost ~]# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS                    NAME
S
5f61c80b1666        tomcat              "catalina.sh run"   47 seconds ago      Up 46 seconds       0.0.0.0:90->8080/tcp     kind
_curran
[root@localhost ~]#
```

Things to be noted.

1. **Firewall needs to be running.**
   $ service firewalld status

```
[root@localhost ~]# service firewalld status
Redirecting to /bin/systemctl status firewalld.service
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled;
   Active: active (running) since Wed 2019-07-31 04:39:10 EDT; 1s ago
     Docs: man:firewalld(1)
 Main PID: 13440 (firewalld)
   CGroup: /system.slice/firewalld.service
           └─13440 /usr/bin/python -Es /usr/sbin/firewalld --nofork --n

Jul 31 04:39:09 localhost.localdomain systemd[1]: Starting firewalld - 
Jul 31 04:39:10 localhost.localdomain systemd[1]: Started firewalld - dy
[root@localhost ~]#
```

The firewall should be up and running.

Next: lets open the port **"90"** on the local system

$ firewall-cmd --zone=public --permanent --add-port=90/tcp

```
[root@localhost ~]# firewall-cmd --zone=public --permanent --add-port=90/tcp
success
[root@localhost ~]#
```

$ firewall-cmd --reload

```
[root@localhost ~]# firewall-cmd --reload
success
[root@localhost ~]#
```

Now, lets try to access the tomcat application from the local machine.
http://<ip add of the docker machine>:90