Ticket-Monster HA Cluster

Objective: This lab with introduce JAVA devlopement with Docker using JBoss, Wildly and a lot more. We will also look at continous intergration and how to update through Docker Compose.<br>

Preparation: Start be navigating to the appropriate lab folder, and make sure you are in part-b. Then, open several terminal windows and a internet browser window.<br>

Outcome: Once done, Ticket Monster will be deployed on your localhost, complete with updates and potenial modifications.<br>

Data Files: Dockerfile, ticket-monster.war<br>

Step 1. Running the images

1. Create a network:

```
$ docker network create mynet
```

2. Start the postgres server container.

```
docker run --name db -d -p 5432:5432 --net mynet -e POSTGRES_USER=ticketmonster -e POSTGRES_PASSWORD=ticketmonster -docker postgres
```

3. Start the Apache httpd + modcluster.

```
docker run -d --net mynet --name modcluster -e MODCLUSTER
```

```
_NET="192.168.172.10." -e MODCLUSTER_PORT=80 -p 80:80 kar
m/mod_cluster-master-dockerhub
```

4. Check `/mcm` (mod_cluster manager). Before starting the Wildfly servers, open `/mcm` that was exposed on port `80` in the previous step[3]

```
#For Linux containers
open http://127.0.0.1/mcm
```

```
#For docker-machine containers
active=`docker-machine active`; open http://`docker-machi
ne ip $active`/mcm
```

> *NOTE: VM users, must manually open a browser inside the VM, enter the URL info.*

5. Click on `Auto Refresh` link.

6. Start the Wildfly server. Execute:

```
docker run -d --name server1 --net mynet rafabene/wildfly
-ticketmonster-ha
```

7. Check at `/mcm` page that Wildfly was registered at modcluster.

8. You can create as many wildfly instances you want. Let’s just add two more for now.

```
docker run -d --name server2 --net mynet rafabene/wildfly
-ticketmonster-ha
docker run -d --name server3 --net mynet rafabene/wildfly
-ticketmonster-ha
```

9. Access the application.

```
#For Linux Containers
open http://127.0.0.1/ticket-monster/
```

```
#For Docker-Machine Containers
active=`docker-machine active`; open http://`docker-machi
ne ip $active`/ticket-monster
```

10. You can stop some servers and check the application behaviour. Execute:

```
docker stop server1
docker stop server2
```

11. Clean up all containers.

```
docker rm -f `docker ps -aq`
```

Step 2. Ways to Update Ticket-Monster

> *NOTE: This is shown here for learning purposes. This approach is not recommended because the changes are not persisted and the updated version will be lost if the container is restarted.*

With the WildFly server you can deploy your application in multiple ways:

- You can use CLI
- You can use the web console
- You can use the deployment scanner

Get into lab06 file

cd to

build Dokcerfile

TAKE NOTE OF .war location

CHANGE NAME BELOW

1. Remember to start the container exposing the port `9990`. Execute:

```
docker run -d --name server1 --net mynet -p 9990 rafabene
/wildfly-ticketmonster
```

Realize that we don’t specify the host port and we let docker assign the port itself. This will avoid port colissions if running more than one WildFly instance in the same docker host. You can

2. Query the docker host port associated with the running WilDFly container by executing:

```
docker port server1
```

3. You can check if the deployment worked by checking the container log:

```
docker logs -f server1
```

Step 3. Update Ticket-Monster Using the CLI

1. You should already be in your appropriate lab folder. In this case you should find yourself in “lab06:material”. Make sure you are by checking your path (pwd). Once you confirm you are, follow along with these commands:

```
$ cd Dockerfiles
$ cd ticketmonster
```

2. Now, let’s create an image from the Dockerfile inside our current folder. Execute:

```
$ docker build -t ticketmonster .
```

3. Check your local registry to make sure the image is available, then spin that image up:

```
$ docker run -it -p 9990 ticketmonster
```

1. If you have a local installation of WildFly, go to it’s `bin/` folder and run `jboss-cli` to connect to the running WildFly docker container. If you don’t have one, build an image from the Dockerfile found at the link below, and spin up a container. Rmemember, use (-p) to expose the appropriate port.

> *NOTE: The usernmame and password credentials were set in the Docker image.*

```
cd $WIDLFY_HOME/bin
```

2. Run the following, once you are in position:

```
./jboss-cli.sh --controller=<DOCKER_HOST>:<HOST_PORT>  -u
=admin -p=docker#admin -c
```

3. You can also use the docker inspect command to get the docker host port for 9990:

```
#For Linux Containers
./jboss-cli.sh --controller=localhost:`docker inspect --f
```

```
ormat='{{$map := index .NetworkSettings.Ports "9990/tcp"}
}{{$result := index $map 0}}{{$result.HostPort}}' server1
` -u=admin -p=docker#admin -c containers
```

```
#For Docker-Machine Containers
active=`docker-machine active`; ./jboss-cli.sh --controll
er=`docker-machine ip $active`:`docker inspect --format='
{{$map := index .NetworkSettings.Ports "9990/tcp"}}{{$res
ult := index $map 0}}{{$result.HostPort}}' server1` -u='a
dmin' -p='docker#admin' -c
```

4. Once that you're connected through jboss-cli, run:

```
deploy <TICKET_MONSTER_PATH>/ticket-monster.war --force
```

Step 4. Update Ticket-Monster Using the Web Console

1. Use the following to create a Dockerfile, build it, and then spin it up, as done previously: `https://github.com/rafabene/devops-demo/blob/master/Dockerfiles/widlfly-admin/Dockerfile#L6-L7`

> *NOTE: The usernmame and password credentials were set in the Docker image.*

2. Go to the container administration web console in a web browser.

3. Log in with the following credentials:

```
username: admin
password: docker#admin
```

4. Go to the “Deployments” tab.

5. Click on “Replace” button.

6. On the “Step 1/2” screen, select the `ticket-monster.war` file on your computer and click “Next”.

7. On the “Step 2/2” screen, click “Next” again.

8. At this moment the new ticket-monster version should be deployed.

Step 5. Update Ticket-Monster Use the Deployment Scanner

To modify the content inside a running WildFly container that already have applications deployer, you will need to mount a volume from the Docker container in the Docker host.

1. In this example we will use the following host directory:

```
~/wildfly-deploy
```

2. First, we will need to start the containers mapping this directory `~/wildfly-deploy` to `/tmp/deploy` inside the container:

```
docker run -d --name server1 --net mynet -v ~/wildfy-depl
oy:/tmp/deploy rafabene/wildfly-ticketmonster
```

3. Then, copy the `ticker-monster.war` to `~/wildfly-deploy`:

```
cp ticket-monster.war ~/wildfy-deploy/
```

4. Finally, execute a `mv` command **inside** the running container to move `/tmp/deploy/ticket-monster.war` to `/opt/jboss/wildfly/standalone/deployments/`:

```
docker exec -it server1 /bin/bash -c 'mv /tmp/deploy/tick
et-monster.war /opt/jboss/wildfly/standalone/deployments/
'
```

Conclusion

Wow! This is an intense lab, and wether you made it all the way of through without confusing or not, doesn’t matter. These are broad strokes and the details are coming. In this lesson, you took your networking skills to the next level, deploy application servers, deployed a Java application, and learned 3 individual ways to update your apllication. Please, take a moment to do a general reset and restore prepare for the next lecture, or lab.