Section 10 - Docker Compose

2 Docker Compose - Basic Commands

Docker Compose - Installation

- On desktop systems like Docker Desktop for Mac and Windows, Docker Compose is included as part of those desktop installs
- Docker Compose is NOT included in the Docker Linux installation packages
- Docker Compose installation instructions for Linux are available here

Notes:

1. *Docker Compose* relies on Docker Engine to work, so make sure you have Docker Engine installed before you start using the *Docker Compose* tool.
2. *Docker Compose* is not a production-grade tool but is ideal for local development and test

Docker Compose - Common commands

- The most common commands are:

```
$ docker-compose up
$ docker-compose down
```

- The `docker-compose up` => create volumes/networks and start all containers
- The `docker-compose down` => stop and remove all containers, remove all networks

> Notes:
>
> - Volumes are NOT deleted with the `docker-compose down` command
> - To delete the volumes use the `docker-compose down --volumes` command

Docker Compose - Example - Start the Services (1)

- Use the `docker-compose up` to start the services defined in the compose file available in the `resources/compose-sample-2/` directory

```
# cd resources/compose-sample-2/
# docker-compose up
Creating network "compose-sample-2_net-db" with the default driver
Creating volume "compose-sample-2_db-data" with default driver
Creating compose-sample-2_postgres10_1 ... done
Creating compose-sample-2_pgadmin4_1   ... done
Attaching to compose-sample-2_postgres10_1, compose-sample-2_pgadmin4_1
...
postgres10_1  | 2019-09-30 08:57:05.364 UTC [1] LOG:  database system is re
pgadmin4_1    | NOTE: Configuring authentication for SERVER mode.
...
```

Docker Compose - Example - Start the Services (2)

- From the output of the `docker-compose up` command we can see the networks, the volumes and the containers that are created
- From a web browser we can access the `<DOCKER_HOST>:8080` to verify the the pgAdmin web application is running and that we can connect to te postgres DB.
- After the initial phase the logs of all the containers defined in the `docker-compose.yml` file are displayed

Docker Compose - Example - Start the Services (2)

- The command is running in the foreground. We can press `Ctrl-C` to stop all the containers and regain control of the shell prompt

```
...
postgres10_1  | 2019-09-30 08:57:05.364 UTC [1] LOG:  database system is r
pgadmin4_1    | NOTE: Configuring authentication for SERVER mode.
...
pgadmin4_1    | [2019-09-30 08:59:12 +0000] [79] [INFO] Booting worker wit
Gracefully stopping... (press Ctrl+C again to force)
Stopping compose-sample-2_postgres10_1 ... done
Stopping compose-sample-2_pgadmin4_1   ... done
...
```

Docker Compose - Example - Detached mode

- Use the `docker-compose up --detach` or `docker-compose up -d` to start the services in the background

```
# docker-compose up -d
Starting compose-sample-2_postgres10_1 ... done
Starting compose-sample-2_pgadmin4_1   ... done
#
```


Docker Compose - Example - logs

- The `docker-compose logs -f` to display the logs from the containers

```
# docker-compose logs -f
...
postgres10_1  | 2019-09-30 09:23:06.825 UTC [1] LOG:  database system is re
...
pgadmin4_1    | [2019-09-30 09:23:08 +0000] [76] [INFO] Booting worker with
...
```

> Notes:
> In this example we have specified the `-f`, `--follow` option to continuously tail the log messages Press to stop displaying the log messages

Docker Compose - Example - Help

- Many of the commands used from the Docker CLI can be also used with the `Docker Compose` tool.
- Use the `docker-compose --help` to display all commands that can be used

```
# docker-compose --help
...
Commands:
  build              Build or rebuild services
  bundle             Generate a Docker bundle from the Compose file
  config             Validate and view the Compose file
  create             Create services
  down               Stop and remove containers, networks, images, and volu
...
```

Docker Compose - Example - common commands

- Some of the most common commands used with `Docker Compose` are:
  1. `docker-compose ps` => List containers
  2. `docker-compose top` => Display the running processes

```
# docker-compose ps
Name            Command                          State      Ports
-------------------------------------------------------------------------
pgadmin4_1      /entrypoint.sh                     Up      443/tcp, 0.0.0.0:80
postgres10_1    docker-entrypoint.sh postgres      Up      5432/tcp

...
# docker-compose top
...
```

Docker Compose - Example - Project name

- Compose uses the current directory name as the project name.
- The project name is used as prefix name for all containers, networks and volumes created from the `docker-compose.yml` file

```
# docker container ls
... IMAGE                 ... NAMES
... postgres:10           ... compose-sample-2_postgres10_1
... dpage/pgadmin4:4.6    ... compose-sample-2_pgadmin4_1

# docker network ls
NETWORK ID     NAME                          DRIVER   SCOPE
...
91e9ac3cccf0   compose-sample-2_net-db       bridge   local
...
# docker volume ls
DRIVER   VOLUME NAME
local    compose-sample-2_db-data
```

Docker Compose - Example - down

- Use the `docker-compose down` command to stop and remove containers, networks and volumes created by `up`

```
# docker-compose down
Stopping compose-sample-2_postgres10_1 ... done
Stopping compose-sample-2_pgadmin4_1   ... done
Removing compose-sample-2_postgres10_1 ... done
Removing compose-sample-2_pgadmin4_1   ... done
Removing network compose-sample-2_net-db
```

> Note By default volumes are not removed

Docker Compose - Example - down --volumes

- Use the `docker-compose down --volumes` to remove also named volumes declared in the `volumes` section of the Compose file and anonymous volumes attached to containers.

```
# docker-compose down -v
...
Removing volume compose-sample-2_db-data
...
```

Docker Compose - Example [cmd summary]

```
# cd resources/compose-sample-2/
# docker-compose up
* access the <DOCKER_HOST>:8080
* Ctrl-C
# docker-compose up -d
# docker-compose logs -f
# docker-compose --help
# docker-compose ps
# docker-compose top
# docker container ls
# docker network ls
# docker volume ls
# docker-compose down -- Note By default volumes are not removed
# docker-compose down -v
```

Docker Compose - Build (1)

- `Docker Compose` can be used also to build our custom Docker images
- I have seen how to build a custom Docker image using a `Dockerfile`
- Now will see hot to use the `Docker compose` and the `Dockerfile` together to build and run our custom Docker images

Docker Compose - Build (2)

- In the following example we see the `build` section of a `docker-compose.yml` file:

```
version: "3.7"
services:
   webapp:
     build:
       context: ./dir
       dockerfile: Dockerfile-alternate
...
```

Notes

- `context:` => Is the relative path of the Docker build context
- `dockerfile:` => Is the name of the Docker File to use for the build process (optional needed only if != default name)
- The `docker-compose.yml` file, `Dockerfile` and the files related to the build context must be located on the same directory

Docker Compose - Build (3)

- Use the `docker-compose up` to build and start the services defined in the compose file
- Use the `docker-compose build` to build all the custom Docker images defined in the compose file

> Notes:
>
> - The first time the build process will take place to create the custom Docker images.
> - After that the custom Docker images will be available in the local cache.
> - The build process will take place again only if the local cache is invalidated

Docker Compose - Build (4)

- To force the re-build process to recreate the custom Docker image we must use the `docker-compose build` or `docker-compose up --build` command.

> Notes
>
> - Documentation Reference available here

Docker Compose vs other CM tools

- We can see how `Docker Compose` can replace more complicated CM tools such as Vagrant and others
- Avoid the complexity of managing a virtual machine environment with multiple VMs
- With `Docker Compose` we have easier way to setup the development environment. The steps required to setup the development environment could be:
  1. Checkout the code `git clone <repo>`
  2. Use the `docker-compose up` to start all services required for the development

LAB

- Ref:
- D_S10_L02_Docker_Compose_Basic_Commands_LAB.md