Docker Compose

Working with set of containers

Docker Compose

- Docker Compose เป็นเครื่องมือที่ใช้สร้าง และ run docker container พร้อมๆ กันหลายๆ container

- Compose เหมาะที่จะนำมาใช้ในการ develop, test, หรือแม้แต่การทำ Continuous Integration (CI) workflow

- ขั้นตอนของการใช้ compose

    1. define environment ด้วย Dockerfile

    2. define services ของ app ด้วย docker-compose.yml

    3. สั่ง run ด้วย docker-compose up

- ดู docker compose file references ได้ที่
  - https://docs.docker.com/compose/compose-file/
  - https://docs.docker.com/compose/reference/

Prepare Source Files

1. สร้าง folder สำหรับเก็บ code

```
$ mkdir composetest
$ cd composetest
```

2. เปิด Visual Studio Code สร้างไฟล์ชื่อ app.py แล้วเพิ่ม code ของ Python ดังนี้

```
from flask import Flask
from redis import Redis

app = Flask(__name__)
redis = Redis(host='redis', port=6379)

@app.route('/')
def hello():
    redis.incr('hits')
    return 'Hello World! I have been seen %s times.' % redis.get('hits')

if __name__ == "__main__":
    app.run(host="0.0.0.0", debug=True)
```

Docker Image

4. สร้าง file ใหม่ชื่อ requirements.txt แล้วเพิ่ม code ดังนี้

```
flask
redis
```

5. สร้างไฟล์ใหม่ ชื่อ Dockerfile แล้วเพิ่ม script ดังนี้

```
FROM python:2.7
ADD . /code
WORKDIR /code
RUN pip install -r requirements.txt
CMD python app.py
```

6. Build image

```
$ docker build -t web .
```

Define Services

6. เพิ่มไฟล์ docker-compose.yml แล้วเขียน script ดังนี้

```
version: '2'
services:
  web:
   build: .
   ports:
    - "5000:5000"
   volumes:
    - .:/code
   depends_on:
    - redis
redis:
  image: redis
```

Define Services

7. เปิด Terminal แล้วใช้ docker compose สั่ง build และ run ด้วยคำสั่ง

   ```
   $ docker-compose up -d
   ```

   note: -d หมายถึงให้ทำงานแบบ detached mode

8. ใช้ curl เพื่อลอง request ไปที่ web server

   $ curl http://localhost:5000

Some Other Commands

- เราสามารถดู process ของ container ที่ run ด้วย docker compose ได้ ดังนี้

  ```
  $ docker-compose ps
  ```

- สั่งหยุดการทำงาน container ของ compose ด้วยคำสั่ง

  ```
  $ docker-compose stop
  ```

- สั่ง ให้กลับมาทำงาน ใหม่ด้วยคำสั่ง

  ```
  $ docker-compose start
  ```

- เราสามารถ run container เดี่ยวๆ เข้าไป ใน compose ได้ด้วยคำสั่ง run ตามด้วยชื่อ service (ที่เรากำหนดไว้ ใน docker-compose.yml) เช่น

  ```
  $ docker-compose run -d -p 5001:5000 web
  ```

- หยุดการทำงานของ compose และ container ทั้งหมด

  ```
  $ docker-compose down
  ```

Notes

- การสั่ง docker-compose ทำงาน จะต้องมีไฟล์ docker-compose.yml อยู่ใน directory เดียวกันเสมอ

- ตอนที่ docker-compose สร้าง container มันจะสร้าง bridge network และ attach container เข้าไปใน network ด้วย

- เราสามารถสั่ง container ที่ทำงานจากคำสั่ง docker-compose up ได้จาก docker ปกติ เช่น ps หรือ rm

- การสั่ง down, docker-compose จะไล่ลบ container ทั้งหมดรวมทั้ง network ที่สร้างขึ้นมา รวมทั้ง container ที่เพิ่มเข้าไปเองทีหลังผ่าน compose ด้วย

Example :- Wordpress & MySQL

```yaml
version: '2'
services:
  db:
    image: mysql:5.7
    volumes:
      - "./.data/db:/var/lib/mysql"
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: wordpress
      MYSQL_DATABASE: wordpress
      MYSQL_USER: wordpress
      MYSQL_PASSWORD: wordpress

  wordpress:
    depends_on:
      - db
    image: wordpress:latest
    links:
      - db
    ports:
      - "8000:80"
    restart: always
    environment:
      WORDPRESS_DB_HOST: db:3306
      WORDPRESS_DB_PASSWORD: wordpress
```