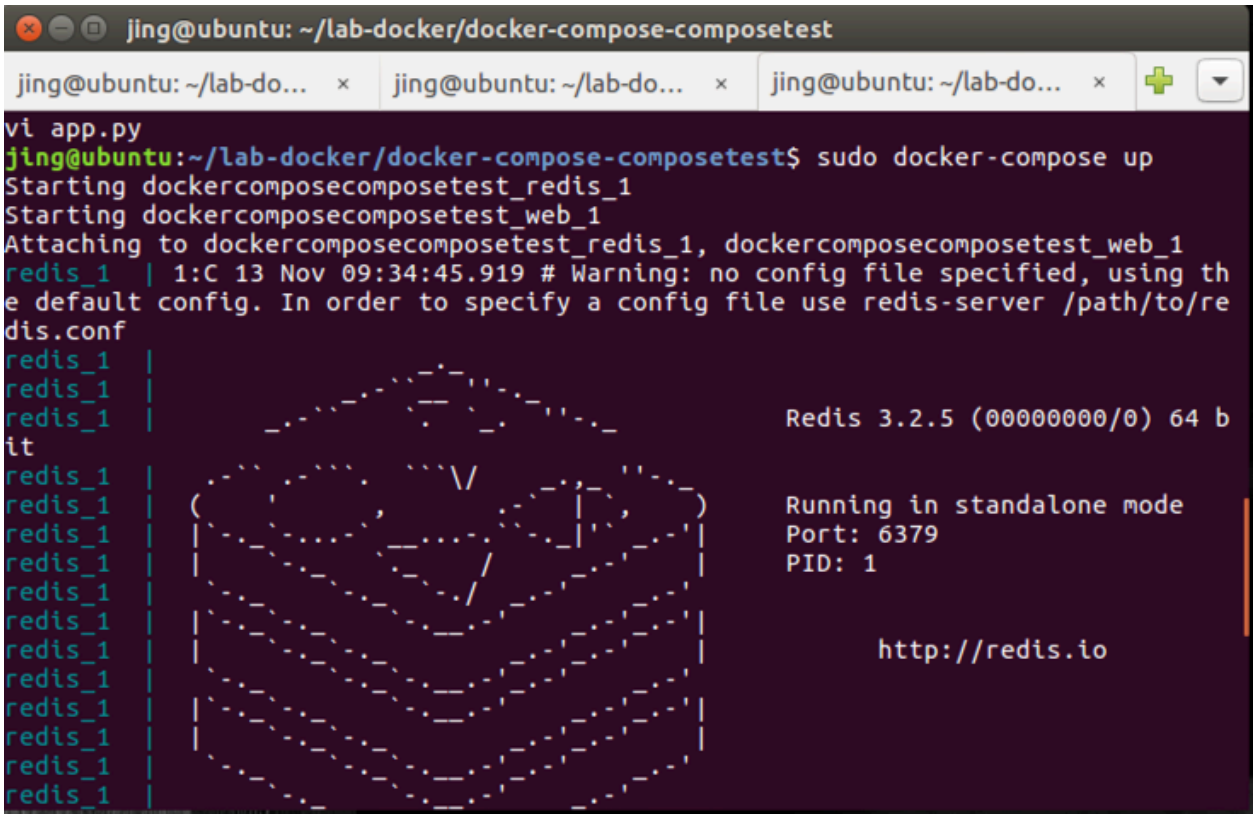
How to use docker-compose, a web app example

Jing, mqjing@gmail.com



ETA: 30 mins

GitHub: Download

Google doc: This document

如果你的 service 是由多個後台 service 所組成的話, 我們可以使用 docker-compose 指令, 把這些 service 安裝並執行在一個一個自己的 container 裡面.

[browser] <---> [your web app] <-----> [redis datastore]

這樣做的好處, 是讓你的 service 執行在乾淨的環境中. 將來使用 swarm 與 docker-machine 時, 就可以輕易的把你的 app, 分散在不同的 host 上執行. 一點也不費力.

下面是一個從官網上節錄修改的例子. 主要是使用 python 搭配 flask 輕量級 framework 撰寫一個 web app, 然後把每次 reload 的次數, 儲存在 redis 伺服器所提供的資料儲存服務上. 每次你用瀏覽器連到 http://localhost:5000 就會自動 count 一次.

好了, 開始吧.

Your Application

app.py

```
from flask import Flask   # import the Python microframe Flask class
from redis import Redis   # import the in-memory datastructure store

# 建立一個以 Flask framework 為基礎的 web app
app = Flask(__name__)

# 連接到 redis 伺服器, 處理中間過程的資料儲存
# host: 會連接到執行在另一個容器中的 service redis_123 伺服器   <---> docker-compose.yml
# port: redis 的 port 是 6379
redis = Redis(host = 'redis_123', port = 6379)



@app.route('/')   # tell Flask the URL '/' will trigger the 'hello' function
def hello():
    redis.incr('hits') # increase the value for the key '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)
```

把你的 app 會用到的套件, 寫在 requirements.txt 裡面. 在這個例子裡面, 我們使用了微型框架 Flask 處理 web app 以及 處理資料結構的 Redis.

requirements.txt

```
flask
redis
```

Create a Docker Image

現在我們來建立一個專屬於 app.py 的執行環境.

Step 1: Create a DockerFile

DockerFile

```
FROM python:2.7    # crate a image based on phthon 2.7
ADD . /code        # add current directory to the new image folder /code
WORKDIR /code      # setup the /code as working directory
RUN pip install -r requirements.txt    # install the necessay pacakge for your app
CMD python app.py                      # execute your app when docker run
```

Step 2: Create Image

Command

sudo docker build -t web •

e.g.

```
jing@ubuntu:~/lab-docker/docker-compose-composetest$ sudo docker build -t web .
[sudo] password for jing:
Sending build context to Docker daemon 4.096 kB
Step 1 : FROM python:2.7
2.7: Pulling from library/python
386a066cd84a: Pull complete
75ea84187083: Pull complete
LibreOffice Calc  Pull complete
                  Pull complete
729baab2cba2: Pull complete
be05488256c3: Pull complete
6721fa748dcf: Pull complete
Digest: sha256:ef77ed1473ac3be3f7143d2007a62ddd63780f316ad1bdff18cd48e495cbc5c3
Status: Downloaded newer image for python:2.7
 ---> d91d61d657d2
Step 2 : ADD . /code
 ---> 5f6992a62f5a
```

把一切串起來, 定義你 app 的 services

首先要啟動 radis 伺服器處理資料的儲存, 接者啟動你的程式提供服務. 我們使用 docker-compose 來處理多個 services 啟動的問題.

下面的 yml 檔會啟動兩個 container, 並且按照相依關係啟動.

docker-compose.yml

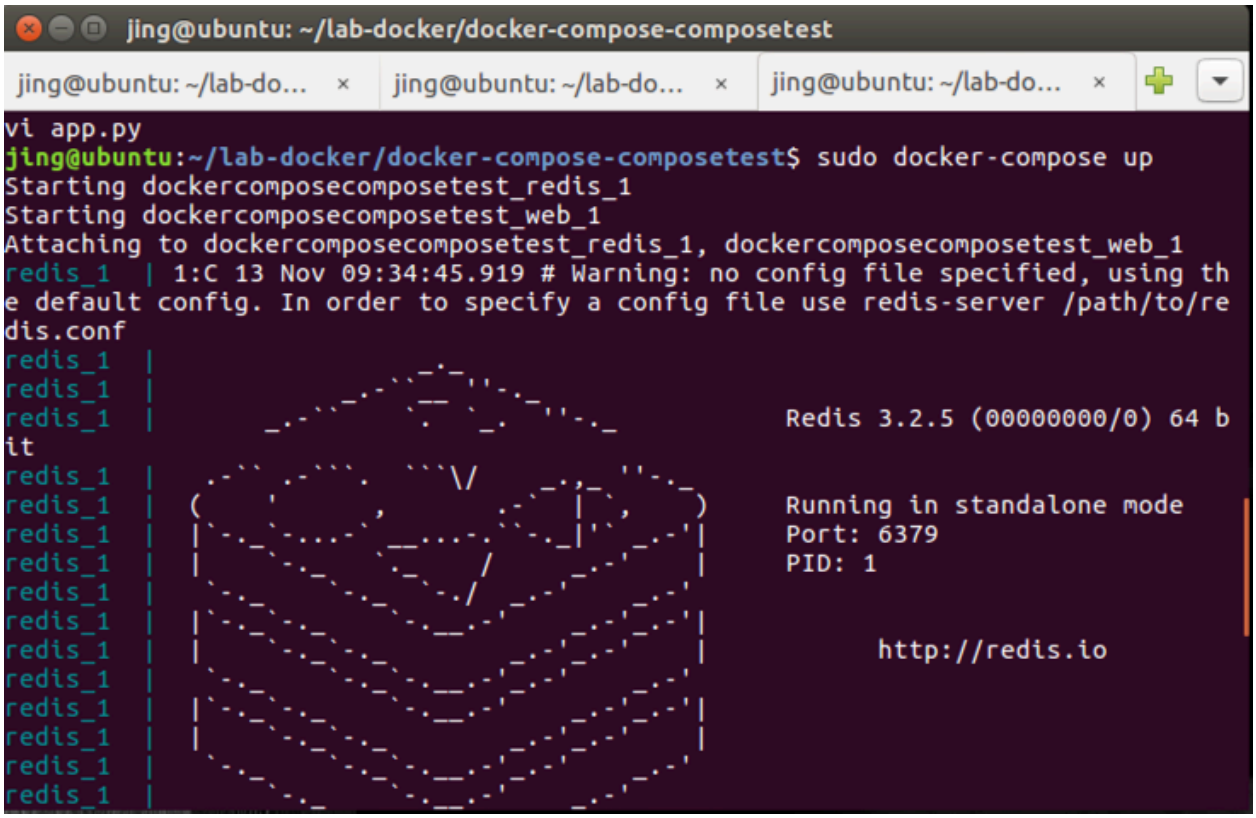
```
version: '2'

# Create two service: web, redis
services:
  web:
    build: .
    ports:
     - "5000:5000"
    volumes:
     - .:/code
    depends_on:    # web service depends on redis service, so redis service will be launch first.
     - redis_123
  redis_123:
```

image: redis　#啟動這個 service 時, 會把 redis image 拉下來. 然後一直 listen 6379 port

Run

sudo docker-compose up

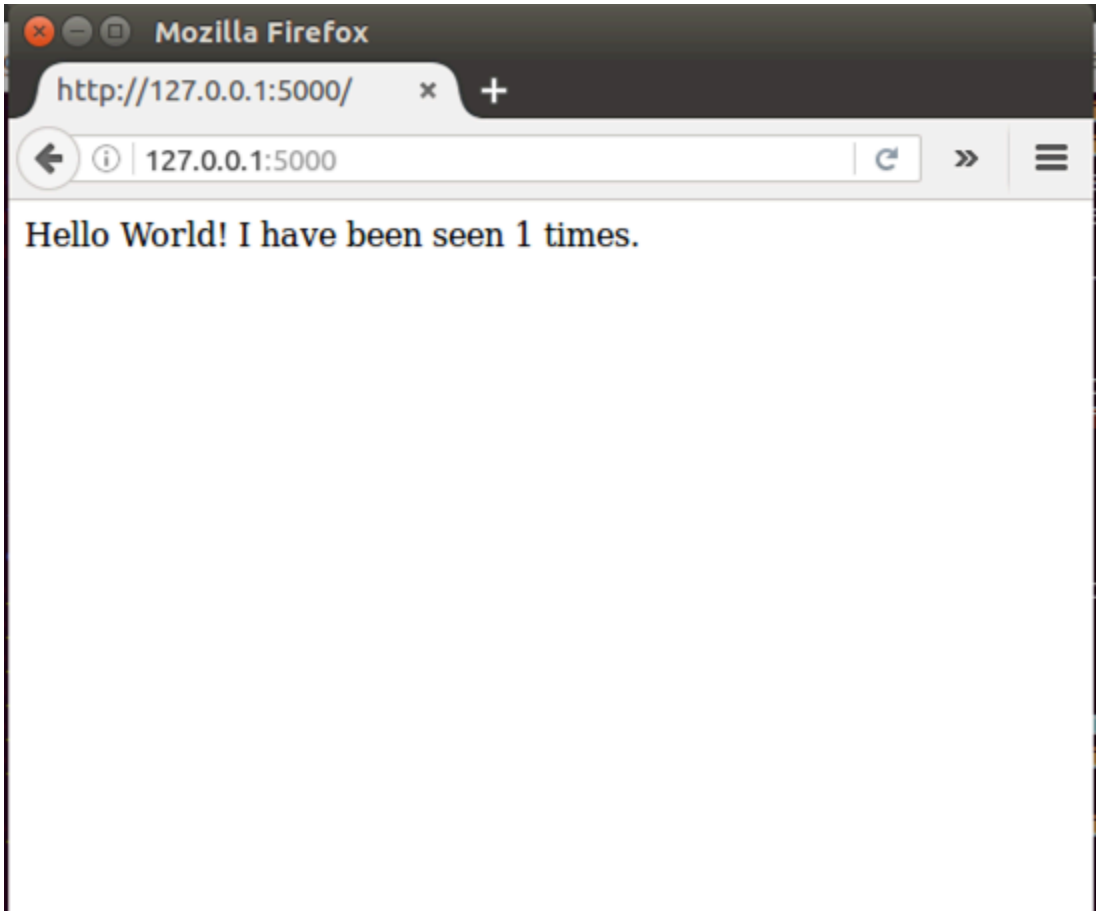


Verification

你的系統裡面會有兩個 container:

1. dockercomposercomposetest_web
2. redis

```
jing@ubuntu:~/lab-docker/docker-compose-composetest$ sudo docker ps
CONTAINER ID        IMAGE                               COMMAND                  CREATED             STATUS
      PORTS                    NAMES
55d4631d2a4b        dockercomposecomposetest_web    "/bin/sh -c 'python a"   47 minutes ago      Up 19 seconds
      0.0.0.0:5000->5000/tcp   dockercomposecomposetest_web_1
9b79084d5cce        redis                               "docker-entrypoint.sh"   47 minutes ago      Up 19 seconds
      6379/tcp                 dockercomposecomposetest_redis_1
```



References

1. https://docs.docker.com/compose/gettingstarted/
2. http://flask.pocoo.org/docs/0.11/quickstart/#url-building
3. https://www.fullstackpython.com/blog/install-redis-use-python-3-ubuntu-1604.html

4.