On the first day of docker

What I learned in my first 8 hours
Rob Charlwood - Bitniftee Ltd

PyData & DBBUG Xmas Special
10th December 2018

bitniftee

What is Docker?

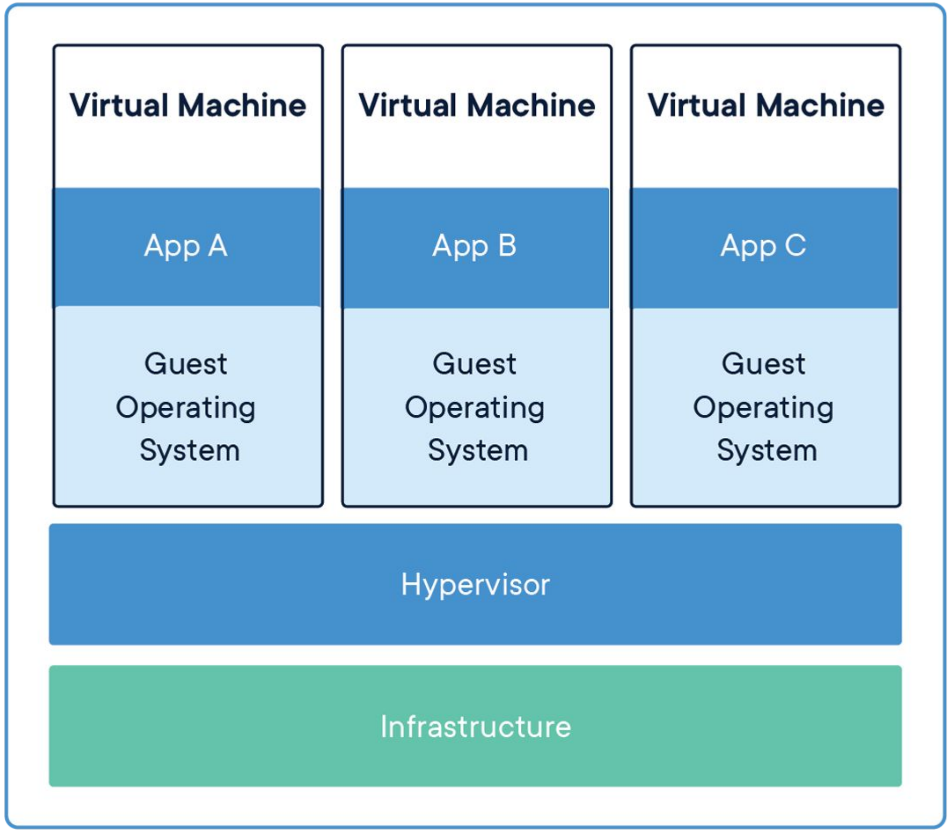
- Docker is software that performs operating system level virtualization - also known as containerization
- Initial release was March 2013
- Docker provides desktop applications for both Mac OSX and Windows (ergh).
- It can also be installed locally on systems running Linux too.

So what exactly is a container?

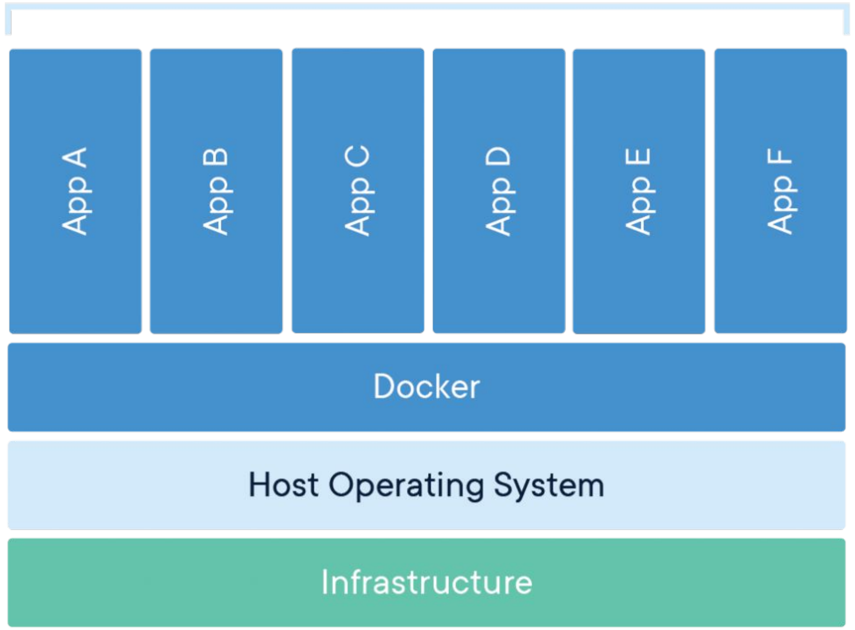
A container is a standardized unit of software

- Packages up code and all its dependencies so the application runs from one computing environment to another. Whether this is locally, in a CI pipeline, staging or production.
- Containers share the host machine's OS system kernel and therefore do not require an OS per application, improving server efficiencies and reducing running costs.

Virtual machines virtualise the hardware whilst containers virtualise the host's operating system. This makes containers portable, small in size and efficient.


Virtual Machine
Virtual Machine
Virtual Machine
App A
App B
App C
Guest Operating System
Guest Operating System
Guest Operating System
Hypervisor
Infrastructure

Containerized Applications



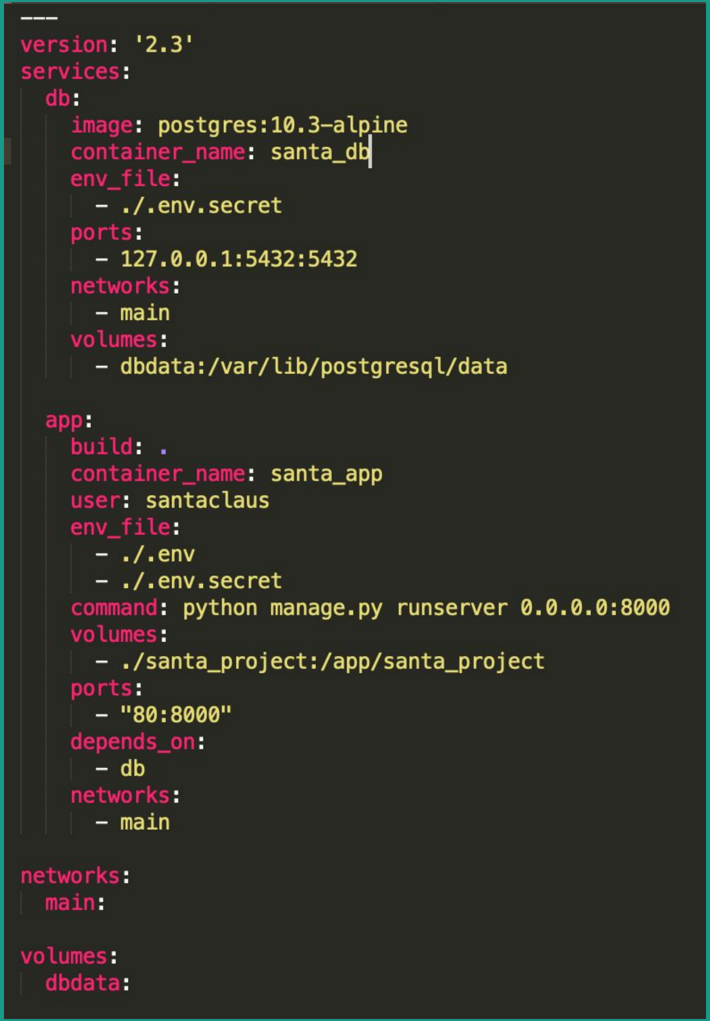
Enough tech jabber, tell me how I can build something!

Our Santa Docker App

- A simple local development system
- Postgres Database
- Application (Python with Django)

Step 1 - Install Docker. :)

Step 2 - Create a ``Dockerfile`` and ``docker-compose.yml`` file.



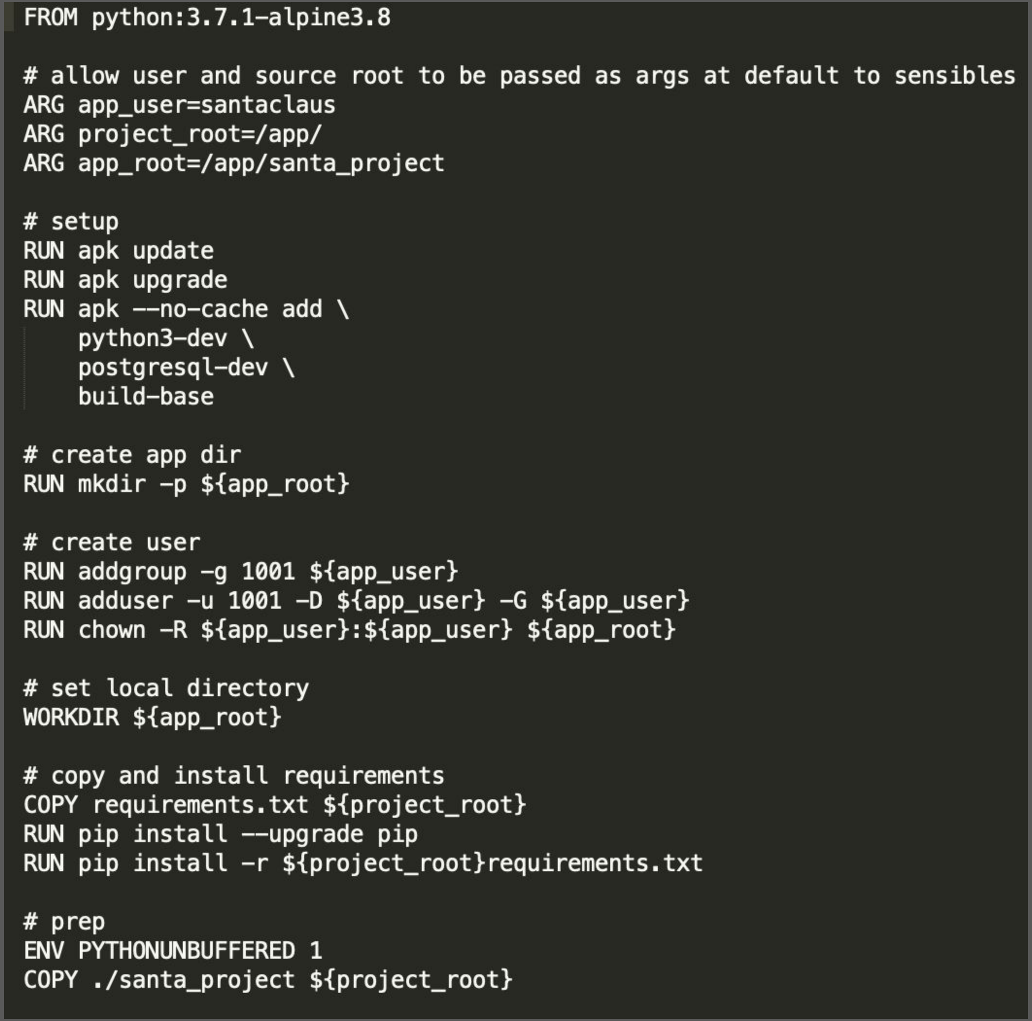
```
---
version: '2.3'
services:
  db:
    image: postgres:10.3-alpine
    container_name: santa_db
    env_file:
      - ./.env.secret
    ports:
      - 127.0.0.1:5432:5432
    networks:
      - main
    volumes:
      - dbdata:/var/lib/postgresql/data

  app:
    build: .
    container_name: santa_app
    user: santaclaus
    env_file:
      - ./.env
      - ./.env.secret
    command: python manage.py runserver 0.0.0.0:8000
    volumes:
      - ./santa_project:/app/santa_project
    ports:
      - "80:8000"
    depends_on:
      - db
    networks:
      - main

networks:
  main:

volumes:
  dbdata:
```

docker-compose.yml



```
FROM python:3.7.1-alpine3.8

# allow user and source root to be passed as args at default to sensibles
ARG app_user=santaclaus
ARG project_root=/app/
ARG app_root=/app/santa_project

# setup
RUN apk update
RUN apk upgrade
RUN apk --no-cache add \
    python3-dev \
    postgresql-dev \
    build-base

# create app dir
RUN mkdir -p ${app_root}

# create user
RUN addgroup -g 1001 ${app_user}
RUN adduser -u 1001 -D ${app_user} -G ${app_user}
RUN chown -R ${app_user}:${app_user} ${app_root}

# set local directory
WORKDIR ${app_root}

# copy and install requirements
COPY requirements.txt ${project_root}
RUN pip install --upgrade pip
RUN pip install -r ${project_root}requirements.txt

# prep
ENV PYTHONUNBUFFERED 1
COPY ./santa_project ${project_root}
```

Dockerfile

Step 3 - ``docker-compose build``

Step 4 - ``docker-compose up -d``

Head to http://localhost

Some other useful commands

- ``docker-compose stop``
- ``docker-compose exec app sh -c “python manage.py makemigrations”``
- ``docker-compose exec app sh -c “python manage.py migrate”``
- ``docker-compose logs -f``
- ``docker-compose logs -f app``
- ``docker-compose logs -f db``

Thanks for listening!

Feel free to ask me questions over a Xmas beer and a mince pie!