虛擬機器 (以作業系統為中心)

虛擬化是將一個應用程式所需執行的環境打包起來, 是系統層上虛擬化

* 透過Hypervisor在一個Host OS上建立多個 Guest OS
* 透過不同的Guest OS, 可以確保整個vm image移植到別台電腦也可以順利運行
* 以作業系統為中心

Docker (以應用程式為中心)

* 改善電腦需要安裝Guest OS 導致啟動慢和佔用大量記憶體的問題
* 作業系統層級上的虛擬化, 透過Container Manager直接將應用程式需要的程式碼和函式庫打包
* 以資源控管機制, 隔離各個容器, 分配 Host OS上的資源
* 不需要Guest OS
* Docker包含映像, 容器, 倉庫
* 倉庫是用來放容器

映像 (Image)

* Image (模子)可以用來產生Container (實體)
* Image可以透過 Dockerfile, DockerHub, Other computer產生
* 唯讀

容器 (Container)

* 是透過Image建立的實體, 每個Container之間都是相互獨立
* 容器在啟動的時候會在最上層建立可以修改的可寫層, 讓容器的功能可以再擴充

倉庫 (Repository) 存在 Registry

* Repository是存放Image的地方,倉庫儲存伺服器 (Registry)存放很多倉庫
* 最大的Registry是Docker Hub, 存放很多 Image

Dockerfile

* FROM node:10.15.3 - 載入Node.js需要的執行環境
* WORK /app – 建立工作目錄 /app
* ADD . /app – 將目前路徑的程式加到工作目錄 /app
* RUN npm install – 運行npm install 透過讀取package.json下載相依的 package
* EXPOSE 3000 – 指定container對外開放的port
* CMD node index.js- 透過 node index.js 執行Server

.dockerignore: 添加 .dockerignore指定忽略 node\_modules

建立Image

* Docker build -t docker-demo-app
* -t : tag
* docker images – 列出全部的 Docker Image

執行成Container

* docker run -p 3000:3000 -it 733776b1db0a
* -p : port
* -it: 透過 docker images得到的 IMAGE ID

通常一個Image會包含很多程式和服務, 但Docker的Image盡量單一化 , 每個Image只包含一個程式, 再由許多 Image堆疊起來, 提高擴充性

* Docker需要一個Base OS , 再使用Base OS的Kernel
* 疊疊樂步驟: 下載Base OS Image -> 執行Image產生Base Container -> 在Base Container安裝新的Image (Server) -> 將整個Container建立新的Image
* **Docker底層實作技術**

透過 Linux Kernel 的Namespace和Cgroup組成

* + Linux Kernel – Namespace

做環境隔離

* + Linux Kernel – Cgroup

做系統資源管理

* **Docker 系統架構**

Client-Server架構 (Docker Client – Docker daemon)

* + Docker Client

用Restful API 連到 Docker daemon

* + Docker daemon

管理Image, 控制Container, 是一個long time service, 提供 Restful API給user操作

* **Docker Image source**
  + From Dockerhub
  + Export from other computer
  + Write Dockerfile by your own
* **Docker Container**

透過 Docker Image實體化的process, 同一個Docker Image可以產生多個Container

* **Registry**

Image都存在Registry, 最大的Registry是 DockerHub, 也可以建立私有的Registry

* 從DockerHub下載ubuntu到local:

**docker search** ubuntu -f is-official-true

% is-official=true: 搜尋官方Image

**docker pull** ubuntu

**docker images**

**docker run** -it ubuntu /bin/bash

% 進入container中的Terminal

**docker ps -a**

* Write Dockerfile
  + FROM Docker Image :版本  
    % 使用到的Docker Image和版本
  + MAITAINER: Quert

% 說明Docker維護的人是Quert

* + RUN

% RUN後面放要執行的Linux命令

* + ADD

% 把local複製到Image裡面

* + ENV

% 環境變數

* + CMD

%CMD ["/apache-tomcat-7.0.82/bin/catalina.sh", "run"]

執行run後就會呼叫TomcatService

* Other
  + Linux

cat /etc/hosts == ifconfig

ifconfig 存在 net-tools之下