Docker介紹與實作

第六組
1032994施奕帆
1033015李奕憲
1032990王偉倫

分工

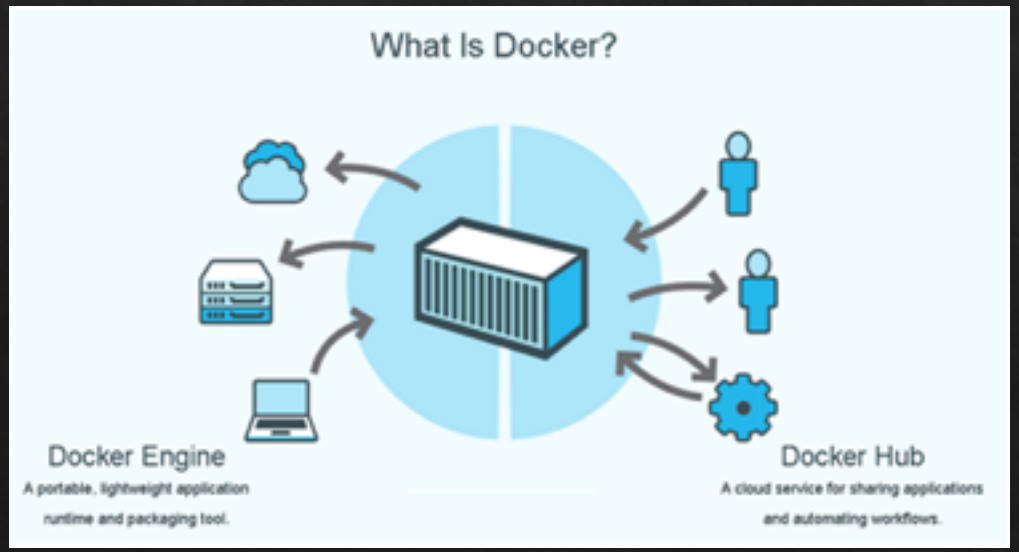
基本介紹　- 李奕憲

比較 & 應用 - 王偉倫

Demo實作　- 施奕帆

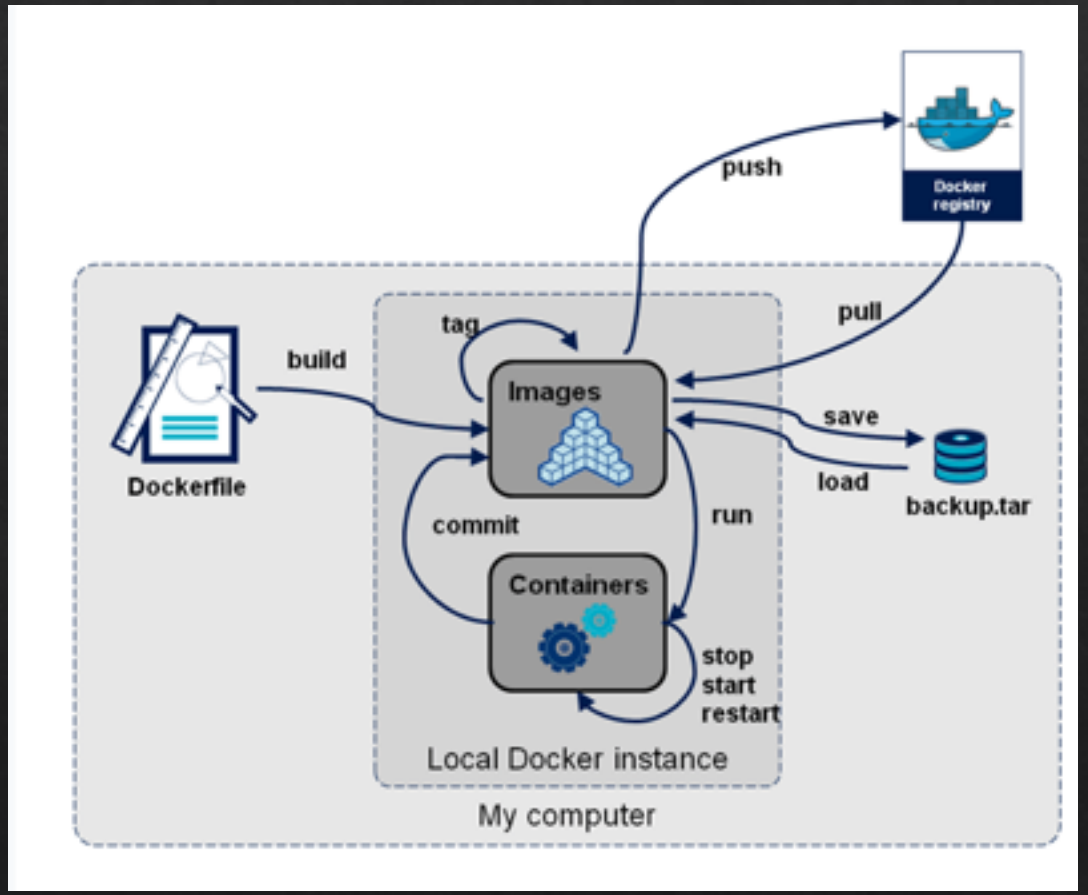
什麼是Docker

- 一個可以包裝應用程式 (OS、Kernel、App) 的開源專案
- 上傳至Docker Hub進行分享 (開發人員、學生)



Docker基本概念

- 映像檔 (Image)
- 容器 (Container)
- 倉庫 (Repository)



映像檔

- 一個唯讀的模板
- 由Docker file建立
- 用來建立容器
- Push上倉庫
- Pull from 倉庫

容器

- 由映像檔產生
- 用來執行軟體
- 每個容器是相互隔離的
- 容器啟動時，在唯讀的映像檔建立一層可寫層作為最上層

倉庫

- 存放映像檔的場所
- Docker Hub
- 工作環境的可攜性
- 分為公開倉庫與私有倉庫

Docker對軟體工程的幫助

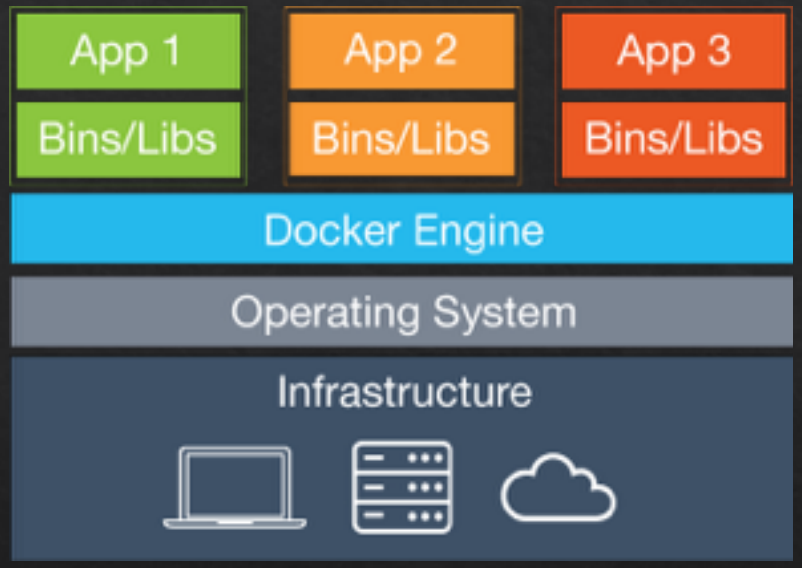
需求分析
系統設計
系統開發
系統測試
系統維護

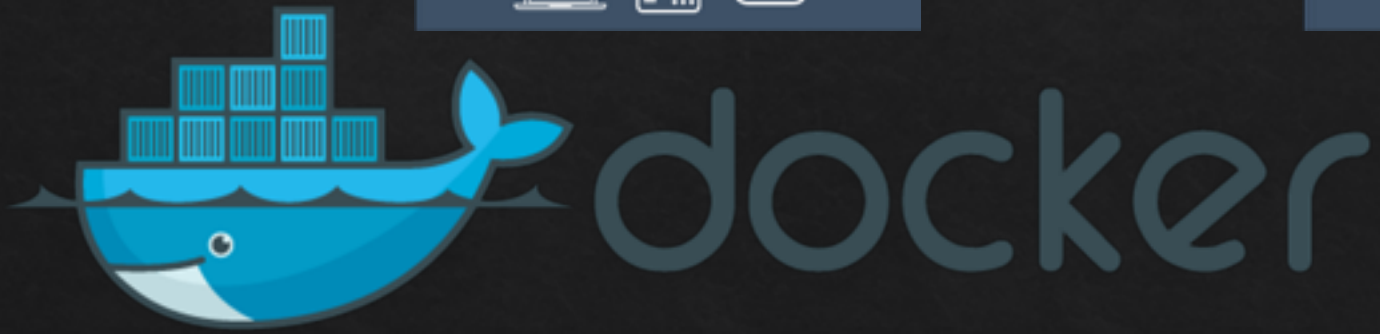
- 改變了軟體生產和軟體交付的方式
  - ➢ 開發人員可以更專注於他們的代碼
  - ➢ 同一個VM運行多種不同的環境
- 開發和運維的緊密配合
  - ➢ 通過Docker file，高效的管理
  - ➢ 分層的文件系統使環境易於管理、配置

Docker與虛擬機的比較

Virtual Machine

Docker





Docker與虛擬機的比較

特性	容器	虛擬機
啟動	秒級	分鐘級
硬碟容量	一般為 MB	一般為 GB
效能	接近原生	比較慢
系統支援量	單機支援上千個容器	一般幾十個

為什麼要用Docker

統一性

問題：以往我們在開發軟體時，常常出現『明明在我的電腦可以跑』

解決：我們通常會使用虛擬機器，來協助統一開發人員, 測試人員, 上線產品的執行環境

低風險

問題：如何降低產品的風險？

解法：縮小每次交付時的變更

導致：更頻繁地交付

與以往VM的差別

- 輕盈感

  你到底有多少時間可以花在coding之外？

- 減少host資源消耗

  你的host又有多少資源可以讓你跑多個VM？

環境部署時間比較

- Virtual Machine
  - 開發環境(VM1), 測試環境(VM2), 上線環境(VM3)
  - Time Wasting:
    - [作業系統安裝（30分鐘）＋部署環境（30分鐘）＋掛載專案（15分鐘）＋開機（5分鐘）]*3 = 4小時

- Docker
  - 開發環境(容器1), 測試環境(容器2), 上線環境(容器3)
  - Time Wasting:
    - 撰寫DockerFile(30分鐘）＋Build成image（30秒）＋[建立容器（30秒）＋掛載專案（30秒）]*3 ＜1小時

Host 消耗資源比較

- Virtual Machine

記憶體： 通常至少2048MB

硬碟： 通常至少6~8G

- Docker

記憶體： 6~10MB

硬碟： 205MB

Docker 如何做到的？

- 效能（指程式運行的速度）
  - 不使用**Hypervisor**，而是直接使用**host**資源
  - 只在可寫層做動作，沒有其他**io**

- 容器啟動速度
  - 少了**vm**開機時的**init**動作

Demo

Demo

你是一位多年都在OSX上工作的工程師，
你今天加入了一個統一在Ubuntu上開發的團隊，
並且這個團隊8小時候要交付給運維人員。

當你已經完成了你的程式碼，
你要怎麼向你的團隊保證你的Code跟他們整合起來不會有問題？
你的團隊又向運維人員保證你們的Code失敗率等於零？

Ans:利用Docker，以及其掛載參數“-v"，將可以快速且消耗少許系統資源實現。

最重要的是：百分之百保證。

檢視本地端映像檔

```
OSX ~ dk images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
kshih/base          latest              2b16b724d921        8 days ago          516.2 MB
centos              web                 4b3649257b3d        3 weeks ago         338.3 MB
centos              centos7             0584b3d2cf6d        8 weeks ago         196.5 MB
```

從Ducker hub下載映像檔

```
 OSX  ~  dk pull kshih/base
Using default tag: latest
latest: Pulling from kshih/base

Digest: sha256:326b5fd0ff649457c6e2910121cc91b80249f75f95378672b54ead250107582a
Status: Image is up to date for kshih/base:latest
```

從映像檔建立容器

建立使用kshih/base的容器

```
OSX ~ dk run -idt -v /Users/kshih/Test:/Volume --name test kshih/base /bin/bash
c3a887165fe55616ae75927b8c08a712ff0f57196e100f6cdb2408ab1b1d7876
```

建立使用Centos的容器

```
OSX ~ dk run -idt -v /Users/kshih/Test:/Volume --name test2 centos /bin/bash
12f0dd1cdfaf9430aa10b93832f3f6427180cea997760a020cbc84f38813c858
```

進入運行中的容器

```
OSX ~/Test dk attach test ✓ 5.61G RAM 2732 22:42:01
root@c3a887165fe5:/#
```

```
OSX ~ dk attach test2 ✓ 5.67G RAM 2734 22:42:22
[root@12f0dd1cdfaf /]#
```

欲掛載的程式碼

```c
#include<stdio.h>
int main(void){

    printf("The Line in Mac\n");
    return 0;
}
```

在兩個Container編譯同一份程式碼

```
root@c3a887165fe5:/Volume# cc test.c -o test.o
root@c3a887165fe5:/Volume# ls
test.c  test.o
```

結果檢測

```
root@c3a887165fe5:/Volume# ls
test.c  test.o
root@c3a887165fe5:/Volume# ./test.o
The Line in Mac
```

結論：

若是兩個Container執行結果都是預期中的

表示此程式碼是通過測試的

END