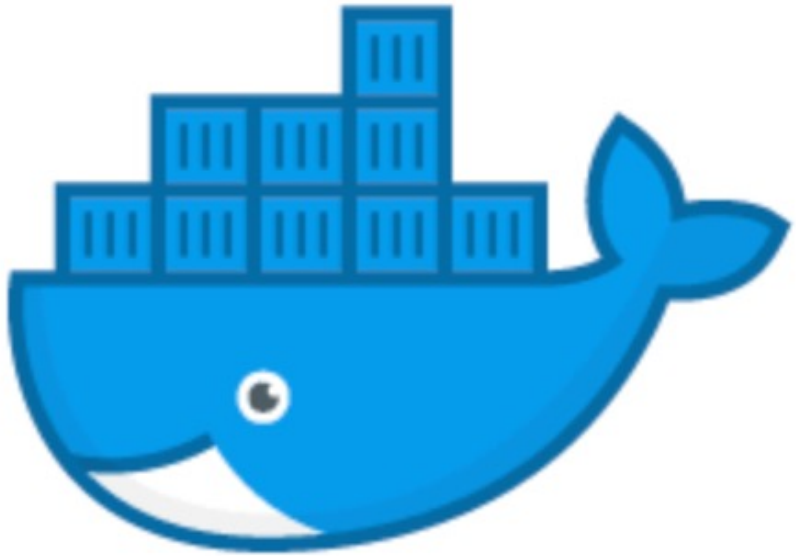


docker

21900806 홍예지

List

01 Docker?

02 Docker Container?

03 VM(Virtual Machine) vs. Container

04 Docker Image?

05 Docker를 쓰는 이유

01 Docker?

[Docker]

by 솔로몬 하익스(Solomon Hykes) - (2013년 3월)

: 애플리케이션을 신속하게 구축, 테스트 및 배포할 수 있는 소프트웨어 플랫폼

-> 소프트웨어를 컨테이너라는 표준화된 유닛으로 패키징하는 기술,
컨테이너에는 라이브러리, 시스템 도구, 코드, 런타임 등 소프트웨어를
실행하는데 필요한 모든 것이 포함되어 있음

Docker는 환경에 구애 받지 않고 컨테이너를 사용하여 응용프로그램을
더 쉽게 만들 수 있도록 설계된 오픈소스 가상화 플랫폼

01 Docker?

[Docker 역할]

- 도커는 리눅스 상에서 컨테이너 방식으로 프로세스를 격리해서 실행하고 관리할 수 있도록 도와주며, 계층화된 파일 시스템에 기반해 효율적으로 이미지, 즉 프로세스 실행 환경을 구축할 수 있도록 해준다

- 도커를 사용하면 이미지를 기반으로 컨테이너를 실행할 수 있으며, 다시 특정 컨테이너의 상태를 변경해 이미지로 만들 수 있다

- 이렇게 만들어진 이미지는 파일로 보관하거나 원격 저장소를 사용하여 쉽게 공유할 수 있으며, 도커만 설치되어 있다면 필요할 때 언제 어디서나 컨테이너로 실행하는 것이 가능하다

02 Docker Container?

[Docker Container]

: 코드와 모든 종속성을 컨테이너에 패키지화하여 응용프로그램이
다른 컴퓨팅 환경에서도 빠르고 안정적이게 실행되는 소프트웨어 표준
(상품을 쉽게 운반하기 위해 사용하는 컨테이너 박스와 비슷한 개념)

컨테이너 박스를 Computer Science에
그대로 적용한 것이 바로
Docker Container

02 Docker Container?

[Container 기술의 특성]

1. 유연성 (Flexible) : 복잡한 애플리케이션들도 모두 컨테이너화 할 수 있음
2. 경량화 (Light Weight) : 컨테이너는 호스트 커널을 활용하고 공유함
3. 변화관리 편의성 (InterChangeable) : 업데이트 및 업그레이드를 즉시 배포 가능
4. 이식성 (Portable) : 로컬로 구축하고, 클라우드와 가상화에도 배치 가능하며, 어디서나 실행이 가능함
5. 확장성 (Scalable) : 컨테이너 복제본을 늘리고 자동으로 배포할 수 있음
6. 스택화 (Stackable) : 서비스들에 대한 수직적 또는 수평적 디자인이 매우 용이함

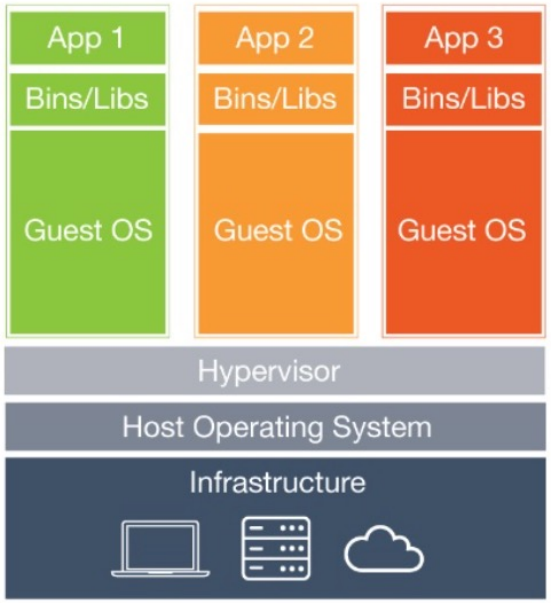
02 Docker Container?

[Docker Container의 역할]

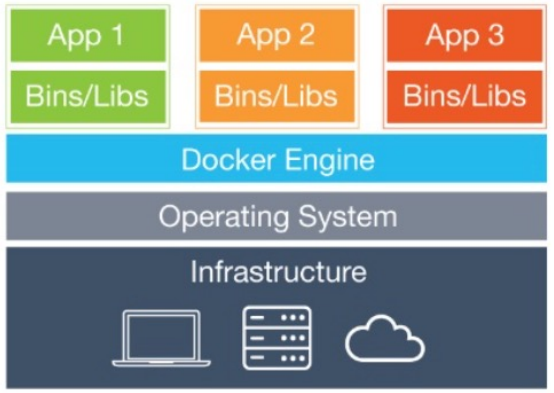
1) 다양한 프로그램, 실행 환경을 컨테이너로 동일한 인터페이스를 제공하여 프로그램의 배포 및 관리를 단순하게 해줌

2) 백엔드 프로그램, 데이터 베이스 서버, 메시지 큐 등 어떤 프로그램도 컨테이너로 추상화 할 수 있고 조립 PC, Google Cloud, AWS 등 어디서든 실행할 수 있도록 해줌

03 VM(Virtual Machine) vs. Container



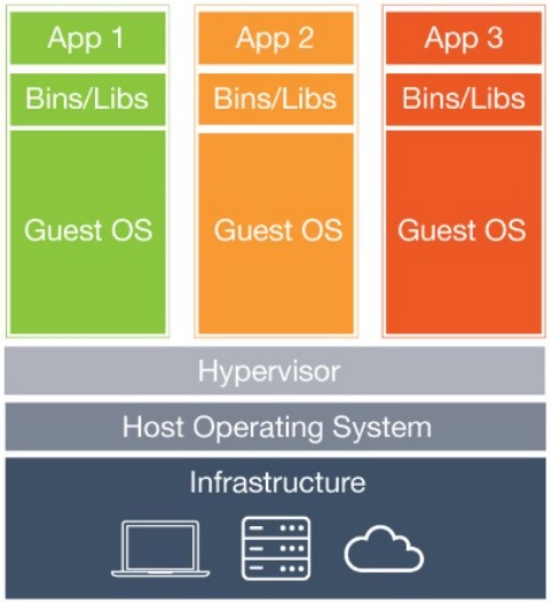
VM



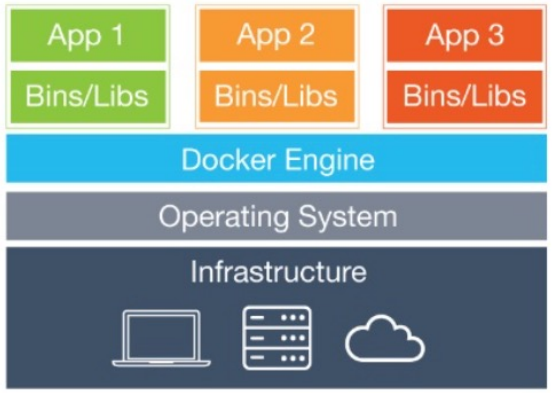
Container

- Window 운영체제를 사용하고 있는데, 리눅스 운영체제를 사용하고 싶을 때
  주로 VM을 사용하여 OS 전체를 가상화 하는 방식으로 사용한다
- 기존 OS인 윈도우와 다른 OS인 리눅스는 독립적으로 존재하게 되어 설치해야 할 용량이 크다
- 또한 다른 OS를 사용하기 위해서는 기존 OS를 이용해야 했기 때문에 속도가 매우 느려진다

03 VM(Virtual Machine) vs. Container



VM



Container

- 하지만 컨테이너는 VM을 사용하지 않고 Docker Engine을 사용하여 동작한다
- 더이상 가상화 하지 않고 개선된 프로세스 격리 방법을 사용
  (-> 별도의 OS 사용이 필요하지 않게 되어 메모리 용량이 작아진다)
- 도커의 이미지는 컨테이너를 실행하기 위한 모든 정보를 가지고 있기 때문에 이것저것 설치할 필요 없이 새로운 서버가 생기면 미리 만들어 놓은 이미지만을 다운 받아 컨테이너를 생성하면 된다

04 Docker Image?

[Docker Image]

: 컨테이너 실행에 필요한 파일과 설정 값 등을 포함하고 있는 것으로
코드 런타임, 시스템 도구, 라이브러리 등 응용프로그램을 실행하는데
필요한 모든 것을 포함하여 독립적으로 실행 가능한 소프트웨어 패키지

Docker Image는 프로그램 실행에 필요한 모든 종속성을
가지고 있으며, 이미지를 사용해서 컨테이너를 생성함
이렇게 생성한 도커 컨테이너를 통해 프로그램을 실행할 수 있다.

04 Docker Image?

[Docker Image 특성]

: Image를 이용하여 여러 개의 같은 컨테이너를 생성할 수도 있음

: 컨테이너를 지워도 Image는 불변한다

: Image들은 실행에 필요한 모든 정보를 가지고 있기 때문에
의존성 파일을 컴파일하거나 따로 설치할 필요가 없다

: 미리 만들어 놓은 Image가 있으면 한 서버에 수십, 수백, 수천 개 실행할 수 있다

Image는 자신이 직접 만들어 관리할 수도 있고, 공개된 Docker Image를
오픈소스 사이트인 Docker hub에서 다운 받아 사용할 수도 있다

05 Docker를 쓰는 이유

Docker는 도커파일(Dockerfile)을 통해 환경 격차를 해결해주는 장점이 있음

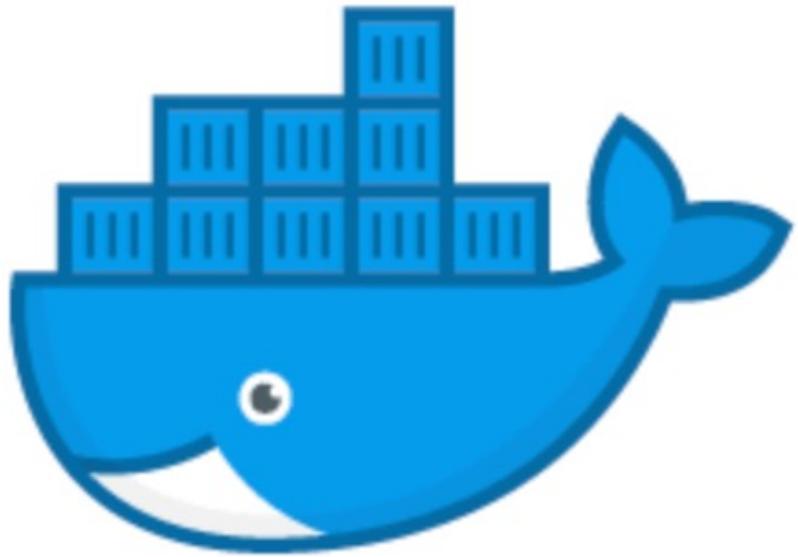
-> 개발환경이 Windows 10 OS인 곳에서 코드를 개발했을 때,
Docker를 사용하면 실제 운영하는 서버의 OS가 Linux OS라도
개발환경에서 실행했던 코드와 동일하게 실행될 수 있도록 해줌

(Docker 컨테이너들은 각각 독립적으로 분리, 격리(isolation) 되어 있기 때문)

05 Docker를 쓰는 이유

기존에 서버마다 서비스와 기능을 나누어 구성했을 경우, 도커를 사용하면
한 개의 서버, 한개의 OS 위에 **여러 개의 도커 컨테이너**를 각각
독립적으로 서비스와 기능을 구현할 수 있게 된다

환경 격차와 격리의 장점이 있는 도커를 사용하면 개발자와 관리자는
환경에 구애 받지 않고 애플리케이션을 **신속하게 배포 및 확장**할 수 있으며
코드가 문제없이 실행되기 때문



docker