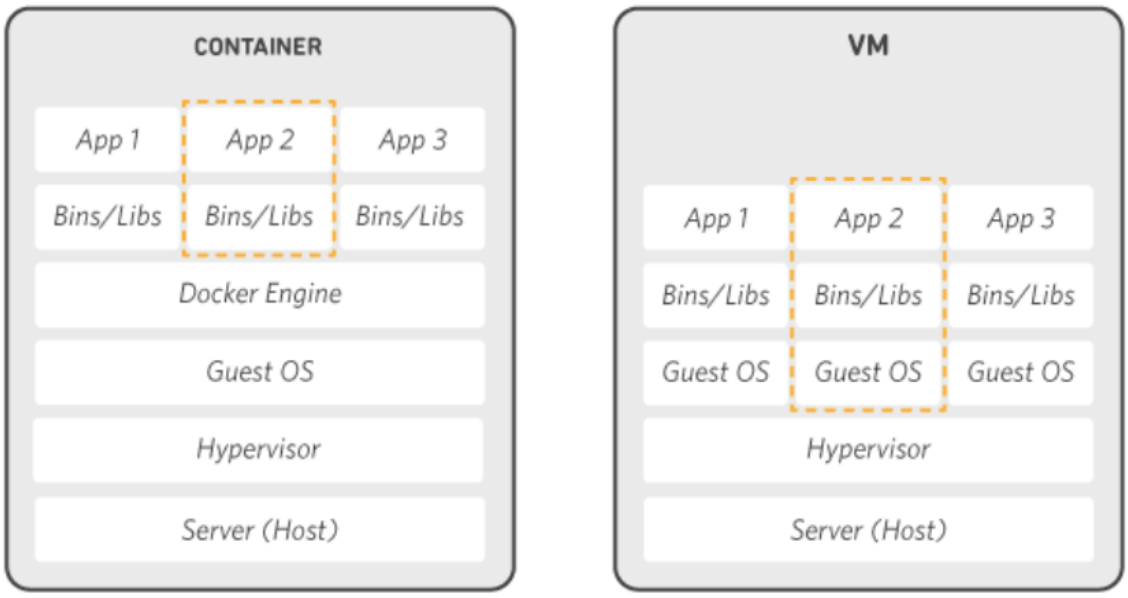
Docker

Docker란?

애플리케이션을 신속하게 구축, 테스트 및 배포할 수 있는 소프트웨어 플랫폼

Docker는 소프트웨어를 컨테이너라는 표준화된 유닛으로 패키징하며, 이 컨테이너에는 라이브러리, 시스템 도구, 코드, 런타임 등 소프트웨어를 실행하는 데 필요한 모든 것이 포함되어 있습니다. Docker를 사용하면 환경에 구애받지 않고 애플리케이션을 신속하게 배포 및 확장할 수 있으며 코드가 문제없이 실행될 것임을 확신할 수 있습니다. AWS에서 Docker를 실행하면 개발자와 관리자가 어떠한 규모에서든 매우 안정적이며 저렴한 방식으로 애플리케이션을 구축, 제공 및 실행할 수 있습니다.

Docker는 코드를 실행하는 표준 방식을 제공합니다. Docker는 컨테이너를 위한 운영 체제입니다. 가상 머신이 서버 하드웨어를 가상화하는 방식과 비슷하게(**직접 관리해야 하는 필요성 제거**) 컨테이너는 서버 운영 체제를 가상화합니다. Docker는 각 서버에 설치되며 컨테이너를 구축, 시작 또는 중단하는 데 사용할 수 있는 간단한 명령을 제공합니다.



Docker의 장점

Docker를 사용하면 코드를 더 빨리 전달하고, 애플리케이션 운영을 표준화하고, 코드를 원활하게 이동하고, 리소스 사용률을 높여 비용을 절감할 수 있습니다. Docker를 사용하면 어디서나 안정적으로 실행할 수 있는 단일 객체를 확보하게 됩니다. Docker의 간단한 구문을 사용해 완벽하게 제어할 수 있습니다. 폭넓게 도입되었다는 것은 Docker를 사용할 수 있는 도구 및 상용 애플리케이션의 에코시스템이 강력하다는 의미입니다.

1. 더 많은 소프트웨어를 더 빨리 제공

   Docker 사용자는 평균적으로 Docker를 사용하지 않는 사용자보다 7배 더 많은 소프트웨어를 제공합니다. Docker를 사용하면 필요할 때마다 격리된 서비스를 제공할 수 있습니다.

2. 운영 표준화

   작은 컨테이너식 애플리케이션을 사용하면 손쉽게 배포하고, 문제를 파악하고, 수정을 위해 롤백할 수 있습니다.

3. 원활하게 이전

   Docker 기반 애플리케이션을 로컬 개발 시스템에서 AWS의 프로덕션 배포로 원활하게 이전할 수 있습니다.

4. 비용절감

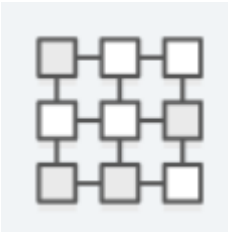
   Docker 컨테이너를 사용하면 각 서버에서 좀 더 쉽게 더 많은 코드를 실행하여 사용률을 높이고 비용을 절감할 수 있습니다.

Docker는 언제 사용하는가?

Docker 컨테이너를 최신 애플리케이션 및 플랫폼을 생성하는 핵심 빌딩 블록으로 사용할 수 있습니다. Docker에서는 손쉽게 분산 마이크로서비스 아키텍처를 구축 및 실행하고, 표준화된 지속적 통합 및 지속적 전달 파이프라인을 통해 코드를 배포하고, 고도로 확장 가능한 데이터 처리 시스템을 구축하고, 개발자를 위한 완전관리형 플랫폼을 생성할 수 있습니다. AWS와 Docker의 최근 협업으로 Docker Compose 아티팩트를 Amazon ECS 및 AWS Fargate에 보다 쉽게 배포할 수 있게 되었습니다.

1. 마이크로 서비스

   Docker 컨테이너를 통해 표준화된 코드 배포를 활용하여 분산 애플리케이션 아키텍처를 구축하고 확장합니다.



2. CONTINUOUS INTEGRATION & DELIVERY

   환경을 표준화하고 언어 스택 및 버전 간의 충돌을 제거함으로써 애플리케이션을 더욱 빠르게 제공합니다.

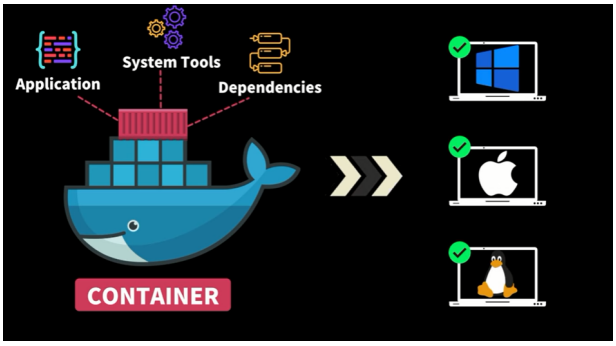
3. 데이터 처리

   빅 데이터 처리를 서비스로서 제공합니다. 데이터 및 분석 패키지를 기술자가 아닌 사용자도 실행할 수 있는 이동식 컨테이너로 패키징합니다.

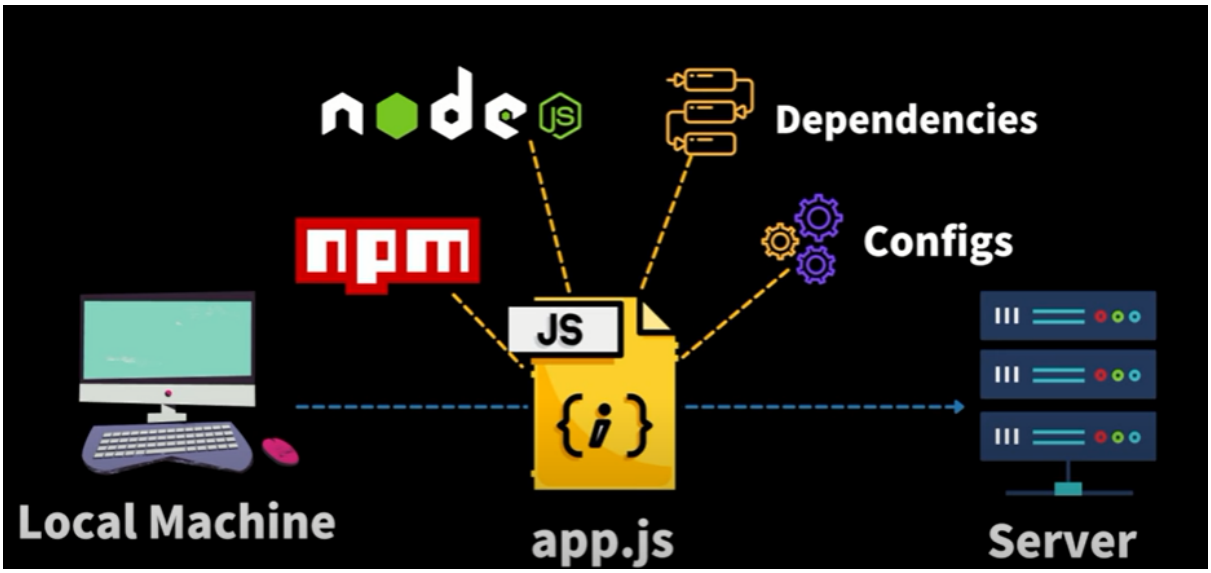
4. 서비스로서의 컨테이너

   안전한 IT 관리형 인프라와 콘텐츠로 분산 애플리케이션을 구축 및 제공합니다.

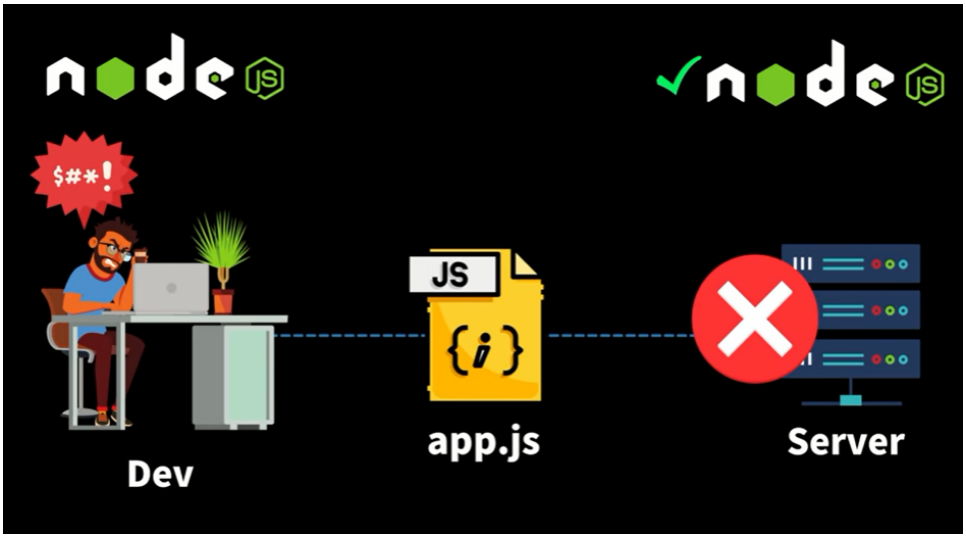
도커 : 컨테이너라고 불리는 하나의 작은 소프트웨어 유닛안에 어플리케이션과 그에 필요한 system tools, dependencies를 하나로 묶어 다른 서버,pc 그 어떤 곳에도 쉽게 배포하고 안정적으로 구동할수있게 도와줌



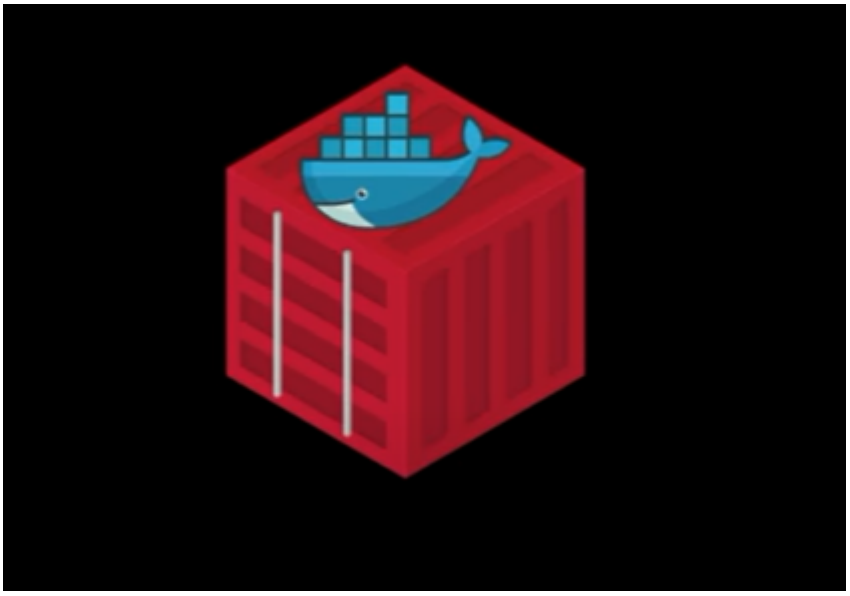
어플리케이션을 구동할 때는 많은 것이 필요합니다. Node js를 예로 들어보면 우리의 소스파일만 서버에 배포하는 것으로는 우리 어플리케이션을 구동하는데 문제점이 있습니다. Node js와 npm 외부 라이브러리를 사용한다면 여러 디펜던시와 환경 변수 이런 모든것들을 다 설정을 해줘야되는데요 서버마다 개발하는 개발자들의 pc마다 이런 모든것들을 설치하고 설정하는 것은 번거롭고 오류도 많이 발생할 수 있습니다.



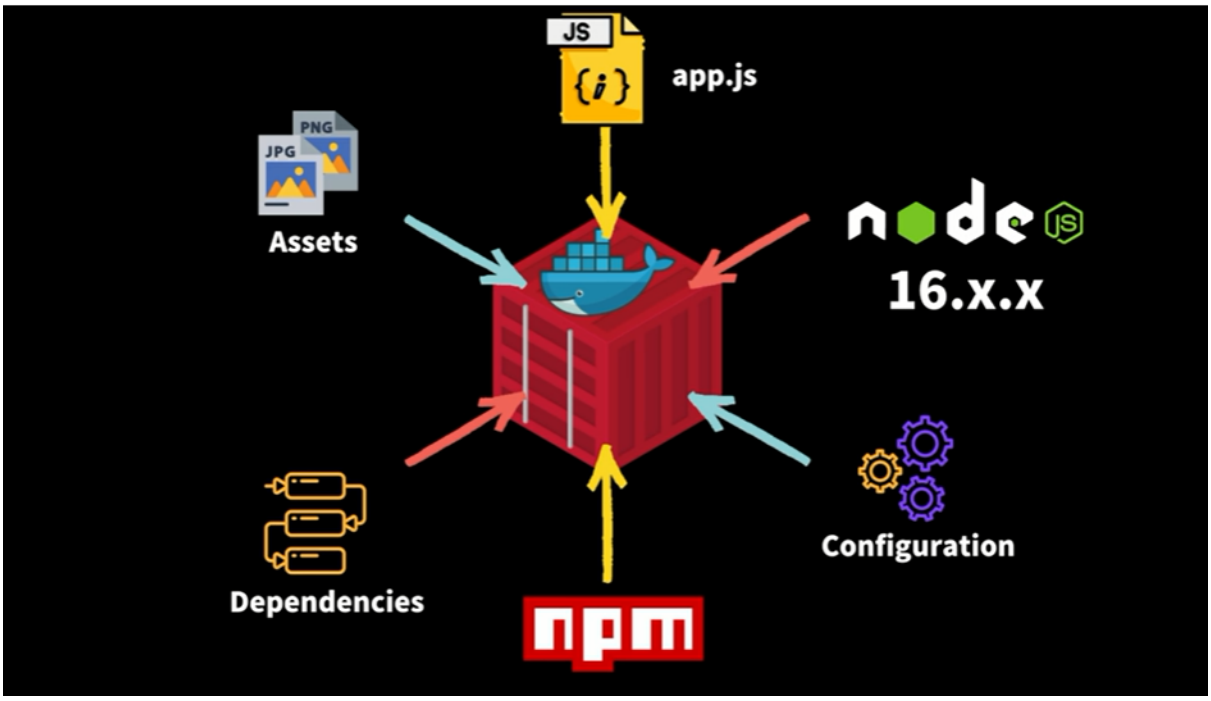
내가 node js를 가지고있고 서버에도 node js가 있으니까 내 소스코드를 서버에 배포하면 자동으로 동작하겠지 라고 생각하다가는 에러가 발생할 수 가 있습니다. 내 pc에서는 잘되는데 왜 서버에서는 안되는거지?라고 생각해보면 node js의 버전이 맞지 않아서 그런걸 수도 있습니다.



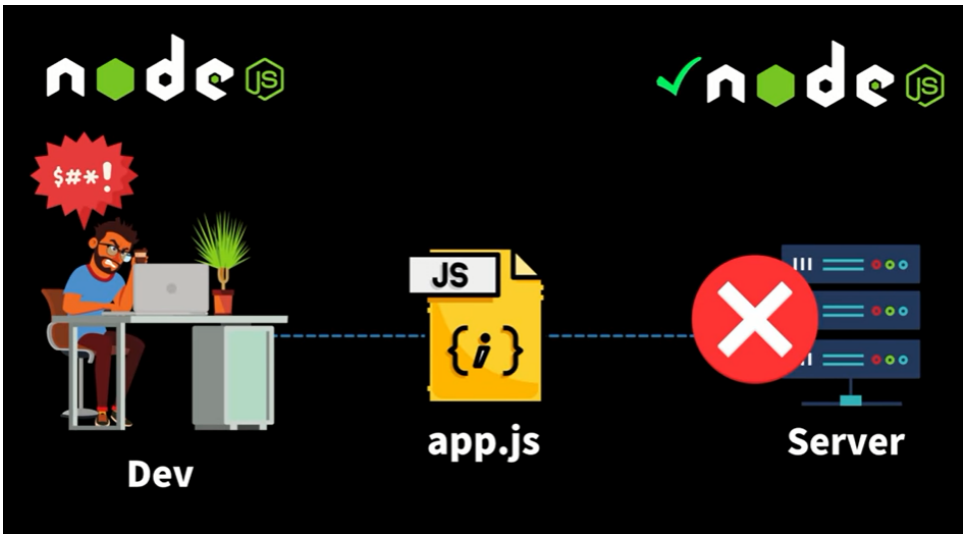
이런 번거로움을 해결하기 위해 도커가 탄생했습니다.



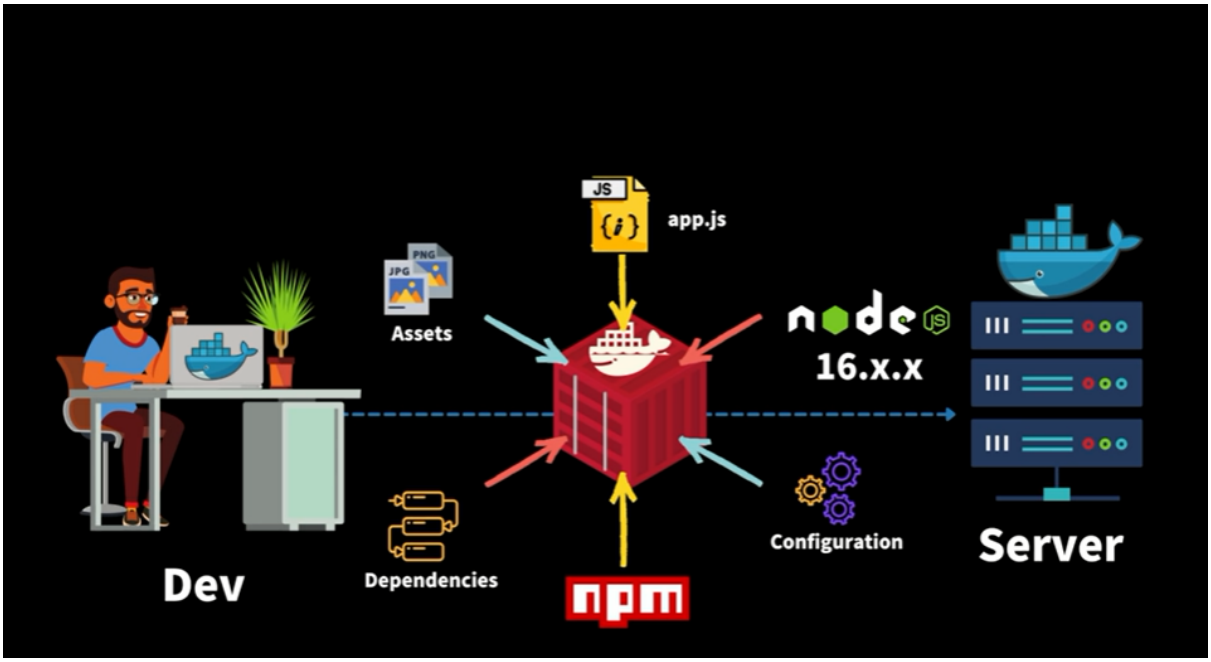
이 도커 컨테이너안에는 어플리케이션 뿐만 아니라 어플리케이션을 정상적으로 동작하게 해주는 node js 환경설정 npm 그리고 여러 라이브러리들의 dependencies 그리고 어플리케이션에 필요한 다양한 리소스들이 포함됩니다. 한마디로 얘기하면 앱을 구동하는데 필요한 모든것들을 이 도커 컨테이너안에 담아놨다 볼 수 있습니다.



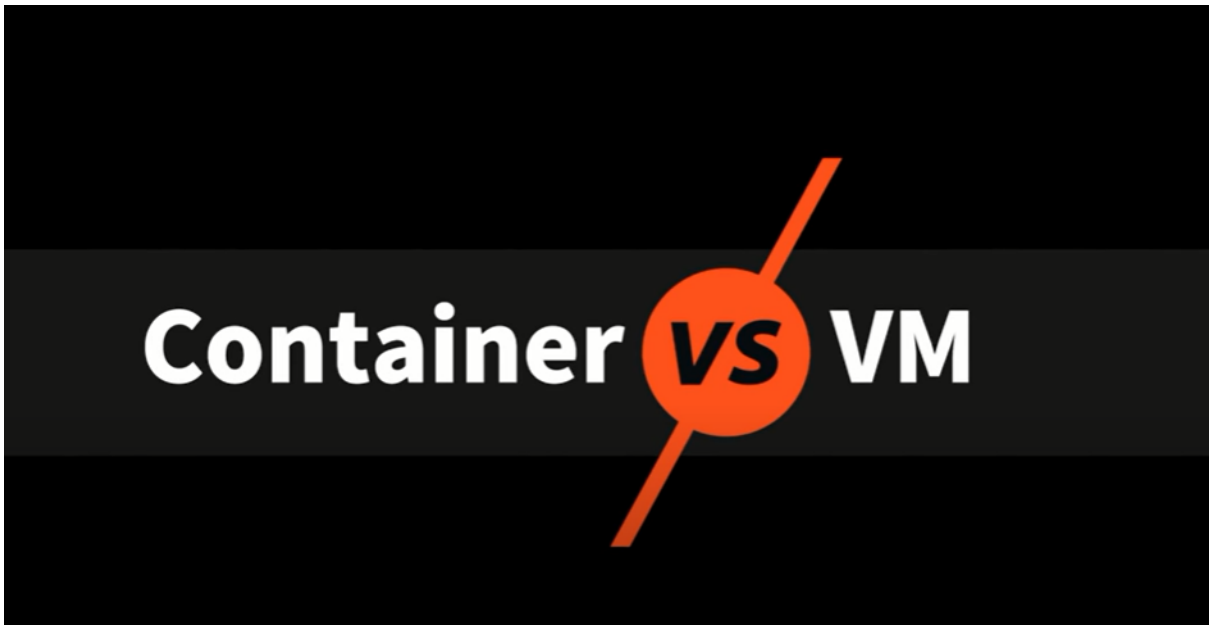
이렇게 도커 컨테이너를 사용하면 앱을 구동하기 위한 런타임환경에 필요한 모든 것들을 어떤 pc에서도 언제든지 동일하게 구동할 수 있습니다. 위와 같은 문제(왜 내pc에서는 작동되는데 니 pc에서는 작동이 안되냐)같은 문제를 해결할수있고 이것저것 설정하고 준비해야되는 문제점을 해결해준다.



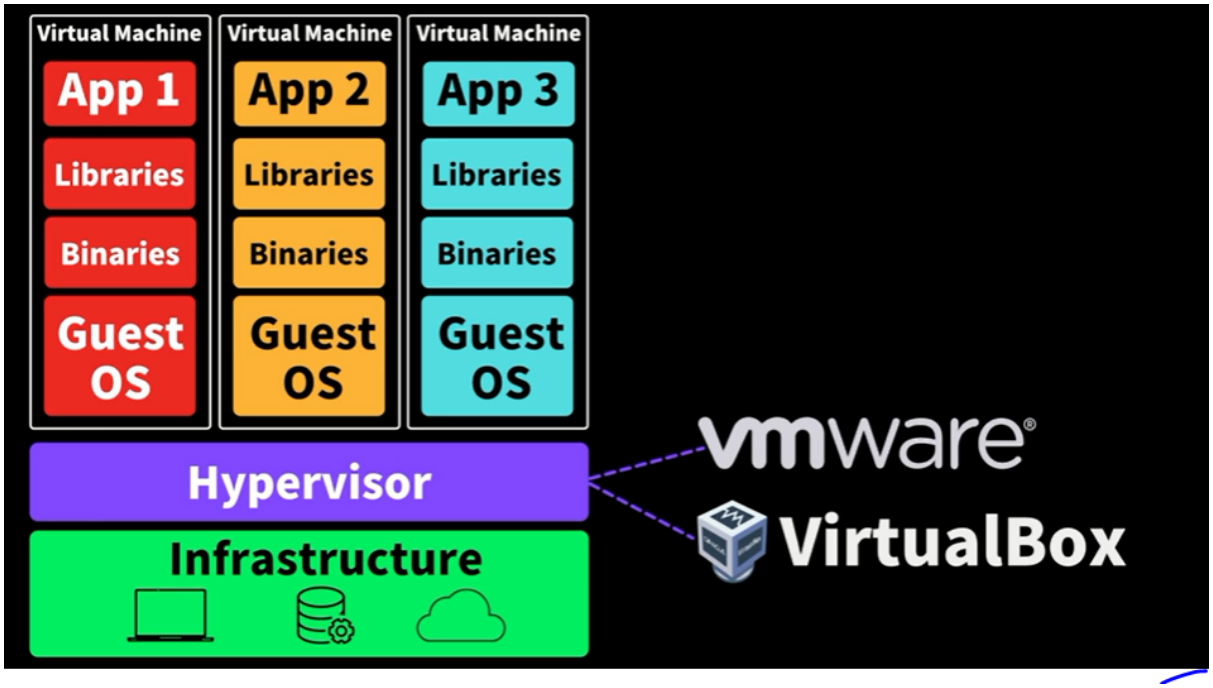
위 사진과 같은 문제가 도커를 사용함으로써 아래와 같이 해결됨



도커의 컨테이너 vs vm



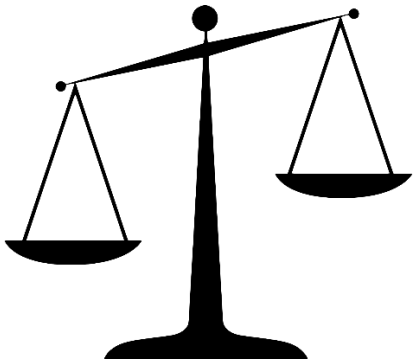
vm은 하드웨어 infrastrucrure위에 vm웨어나 virtualbox같은 hypervisor 소프트웨어를 이용해서 각각의 가상머신을 만든다. 한 운영체제위에서 동일한 어플리케이션을 각각의 고립된 다른 환경에서 구동하기위에서는 vm을 이용해서 어플리케이션을 구동해야한다. 예를들어 mac에서 vm을 이용해 윈도우와 리눅스를 동시에 구동할수있다.

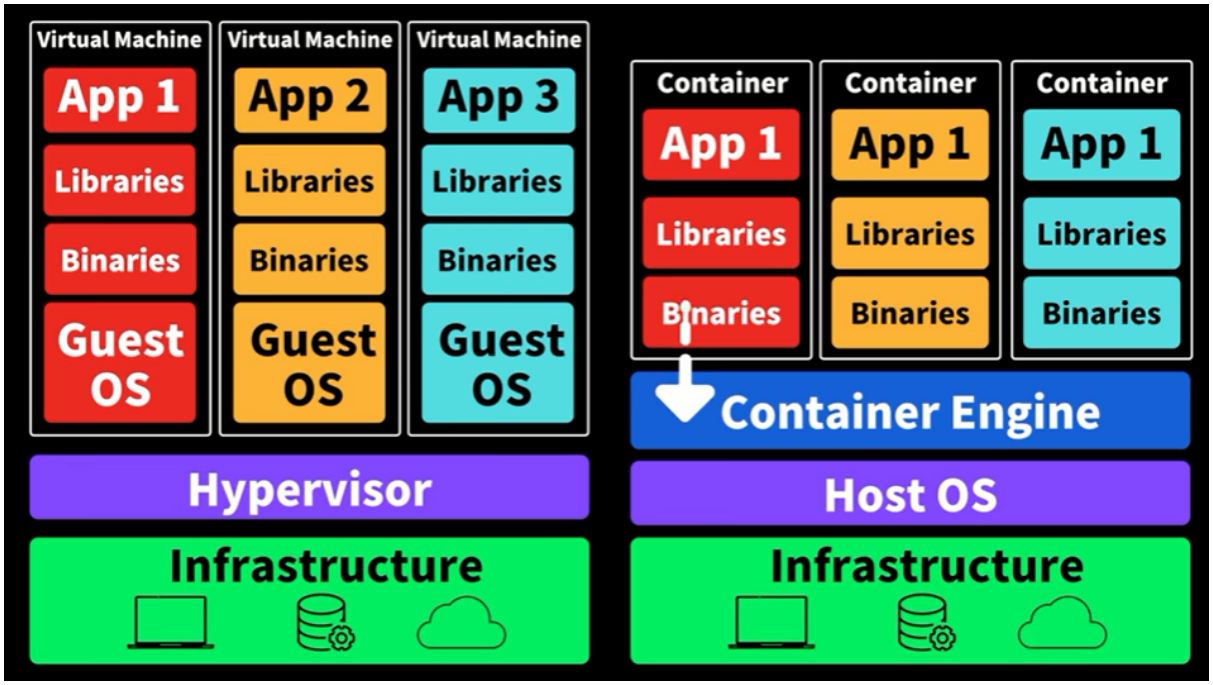


따라서 엄청나게 무거움

운영체제를 포함하고 있기 때문에 굉장히 무겁고 시작하는데도 오래걸리고 infrastructure의 리소스를 많이 잡아먹는 범인이 될수도 잇음.

경량화된 컨셉이 컨테이너임





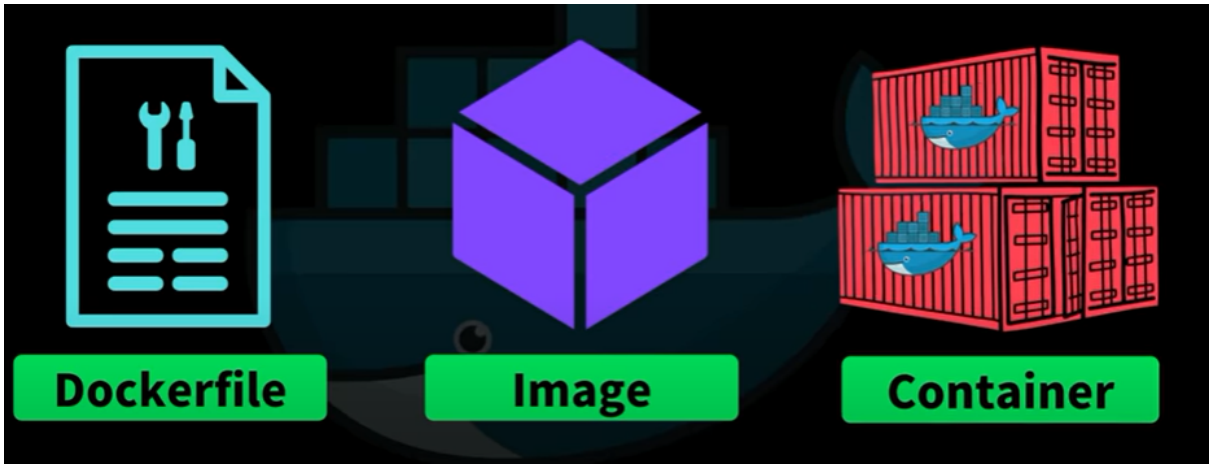
컨테이너는 host os에서 container engine이라는 소프트웨어를 설치만하면 개별적인 컨테이너를 만들어서 각각의 어플리케이션을 고립된 환경에서 구동할 수 있게 해줌

vm과 큰 차이점은 vm은 have 운영체제, 컨테이너는 have not 운영체제 -> 컨테이너 엔진이 설치된 host os를 공유

컨테이너가 구동되기 위해서는 컨테이너엔진이 필요함. 컨테이너 엔진이 호스트오에스에 접근해서 필요한것들을 처리함

컨테이너엔진중에 가장 많이 사용되는 것이 도커

도커는 컨테이너를 만들고 구동하고 배포한다

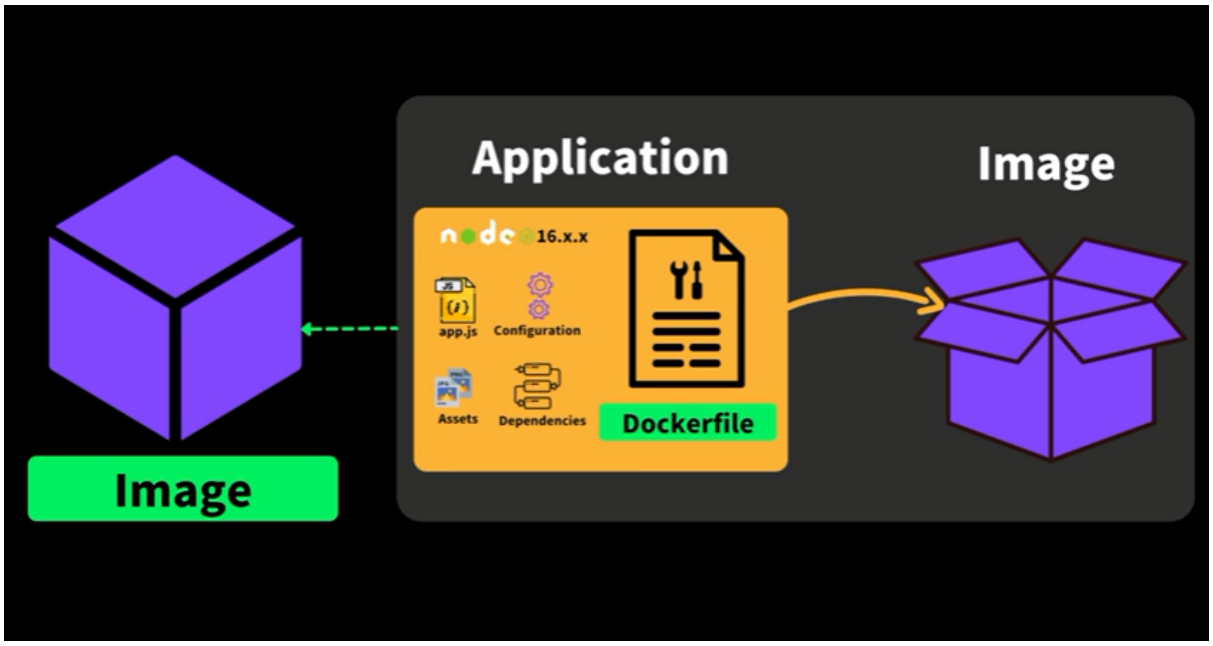


컨테이너를 만들기 위해서는 총 3가지가 필요하다

1. 도커파일을만들고 도커파일을 이용해
2. 이미지를 만들고
3. 컨테이너를 구동함

도커파일은 컨테이너를 어떻게 만드는지 설명서같은거임. 어플리케이션을 구동하기 위해서 꼭 필요한 파일은 무엇이 있는지, 어떤 라이브러리를 설치해야되는지 외부 dependencies에 대해 명시할수 있고 필요한 환경변수에 대해 설정, 어떻게 구동해야되는지 script 도 포함할수잇음

이렇게 작성한 도커파일로 이미지를 만듬



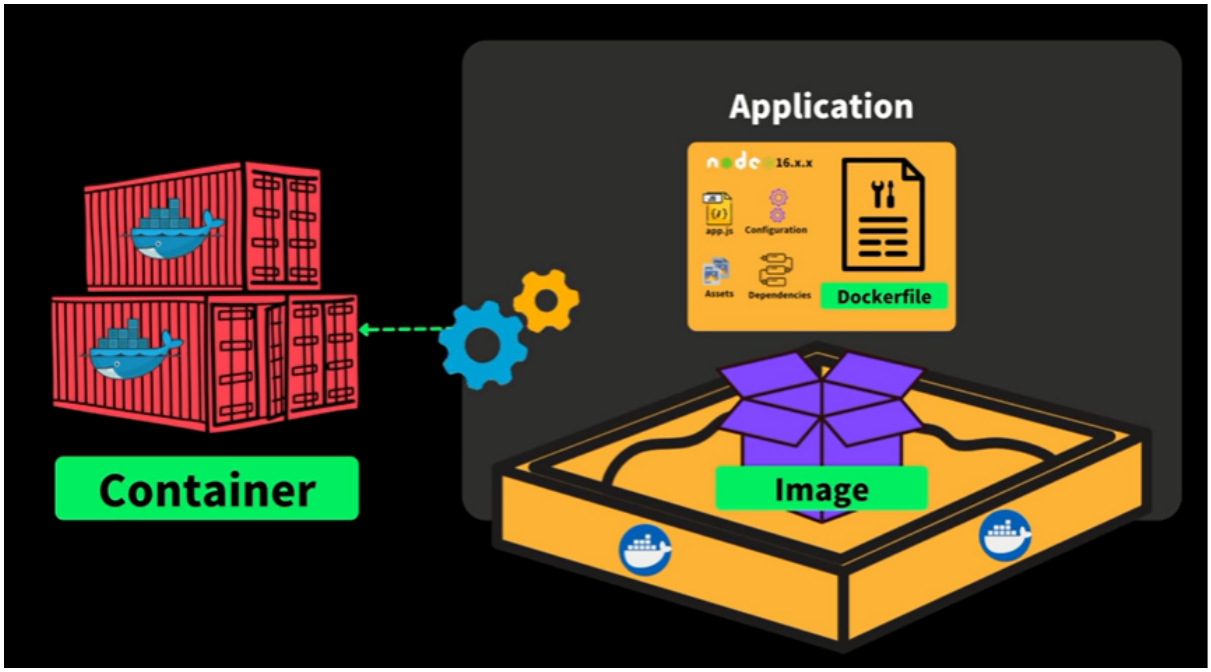
이미지안에는 어플리케이션을 실행하는데 필요한 코드, runtime환경, 시스템툴, 시스템라이브러리

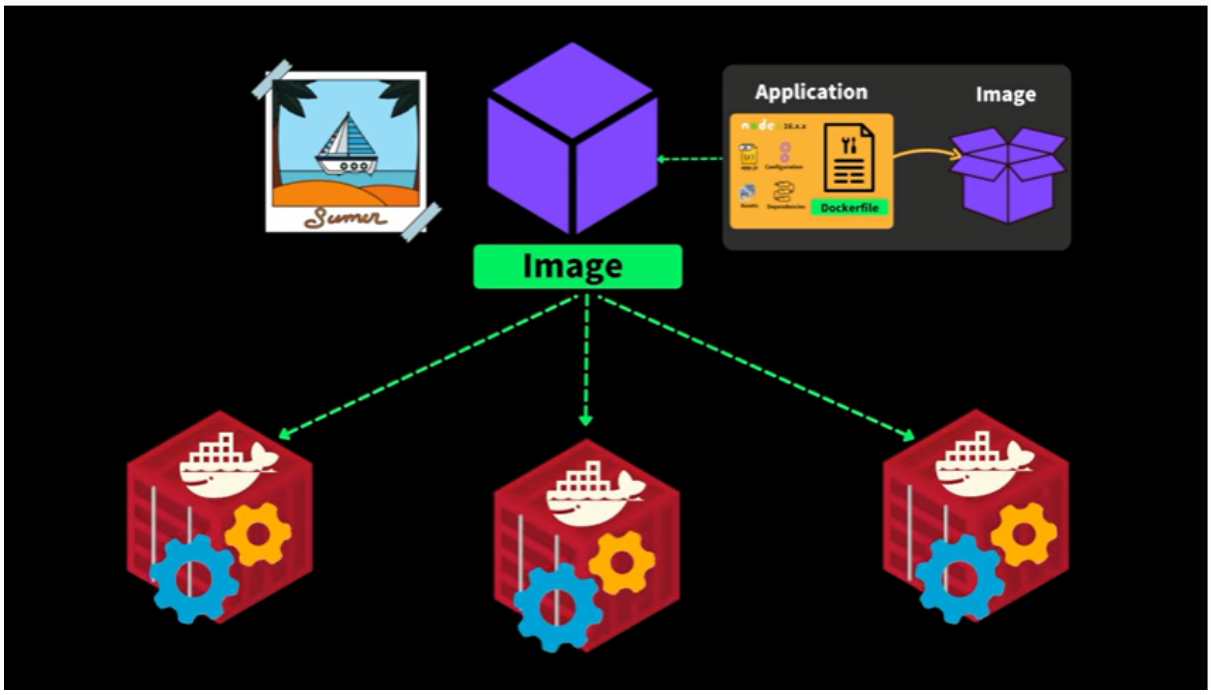
모든 세팅들이 포함되어 있습니다. 한마디로 얘기하면 실행되고 있는 어플리케이션의 상태를 찰칵해서 이미지로 만들어둔다고 생각하면 좋음

만들어진 이미지는 변경이 불가능한 불변의 상태임

마지막단계인 컨테이너는 샌드박스처럼 우리가 잘 캡쳐해둔 우리 어플리케이션의 이미지를 고립된 환경에서 개별적인 파일 시스템안에서 실행하게 할 수 있는 것을 말함

컨테이너 안에서 우리 앱이 동작함 , 고로 컨테이너는 이미지를 이용해서 어플리케이션을 구동하게 되는것임



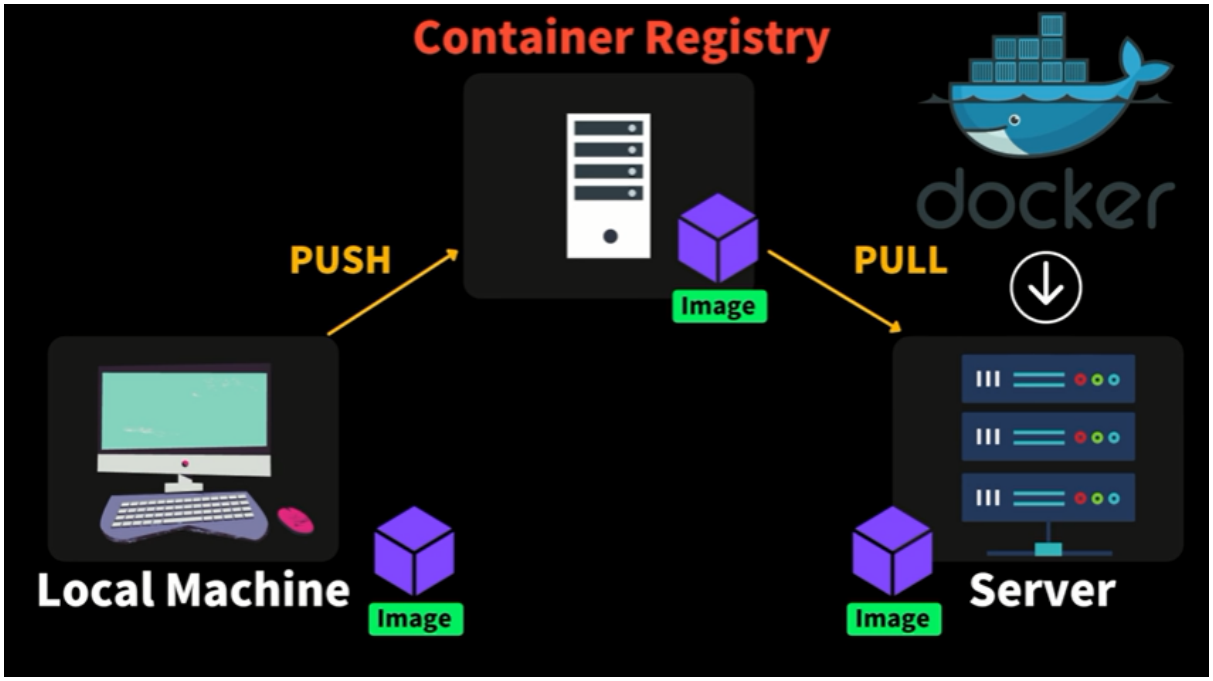


이미지는 불변의 상태, 컨테이너는 개별적으로 수정가능 (이미지에 영향x)

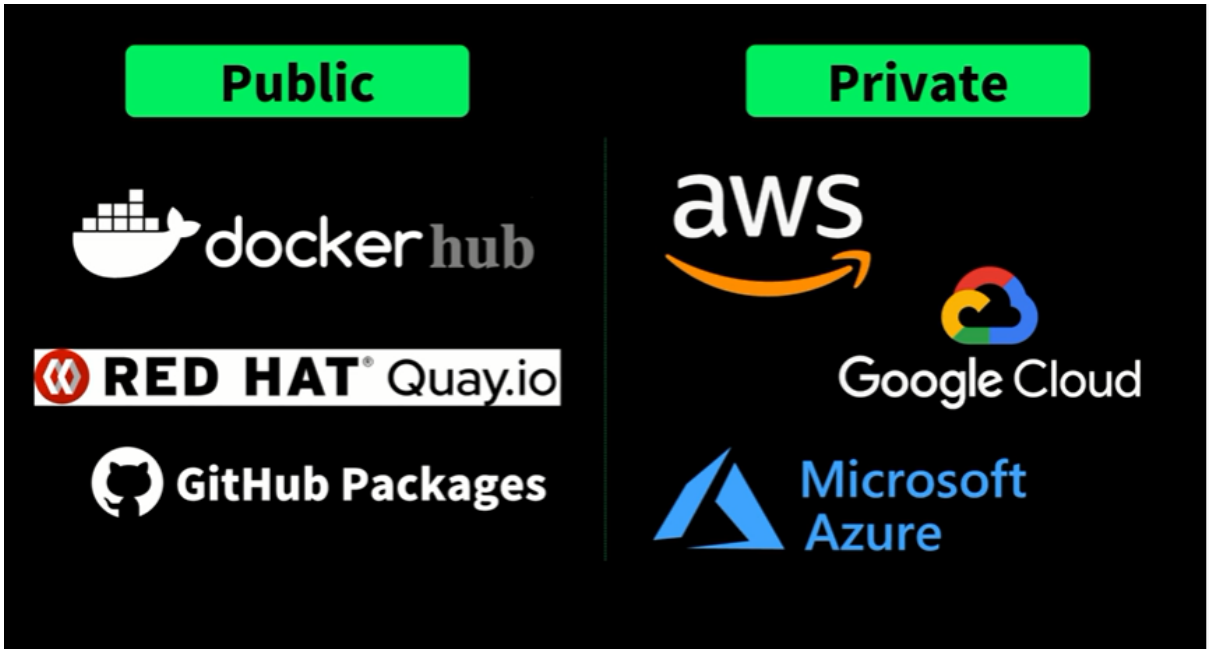
어떻게 컨테이너를 배포하나? (어떻게 이미지를 배포하나?)

로컬머신에서 이미지를 만들어서 container registry에 내가 만든 이미지를 push하고 필요한 서버나 다른 개발자 pc에서 내가 만들어둔 이미지를 가져와서 그걸 그대로 실행하면 됨

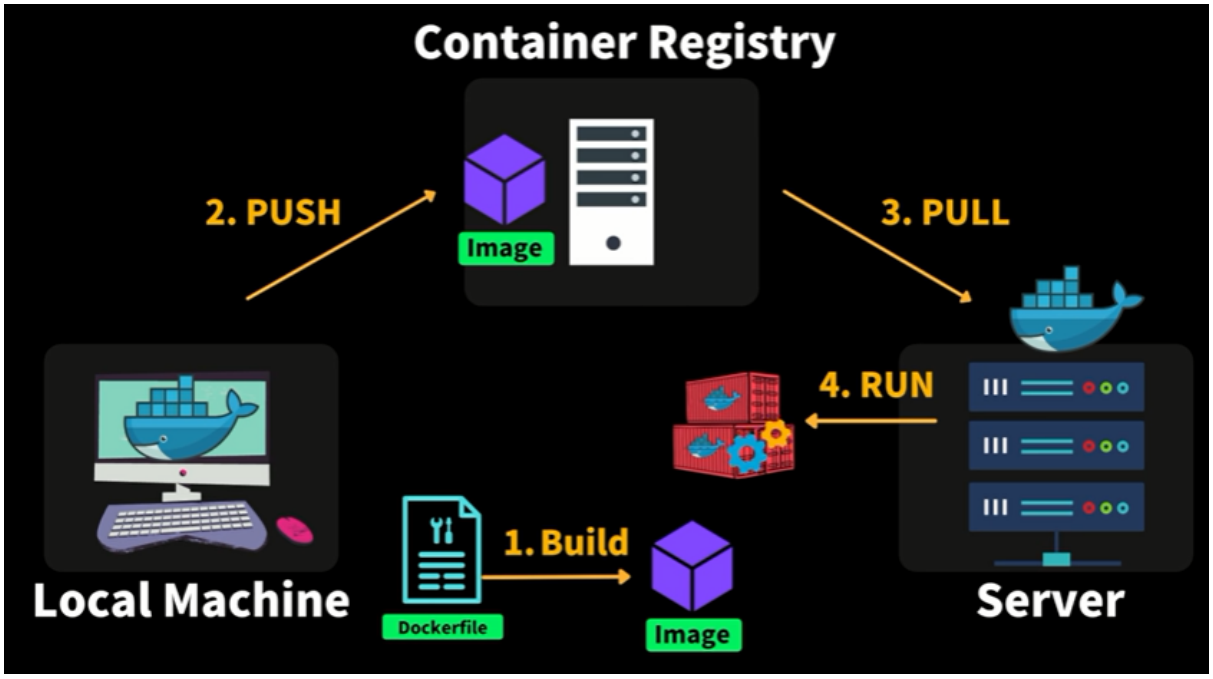
도커와 같은 컨테이너 엔진을 꼭 설치해둬야됨



container registry



총정리



로컬머신에 도커 서버에도 도커를 설치

1. 구동하는데 필요한 도커 파일을 작성
2. 이걸 이용해 이미지를 만듬
3. 이미지를 container registry에 push
4. server에서 다운로드 받아서 container를 실행