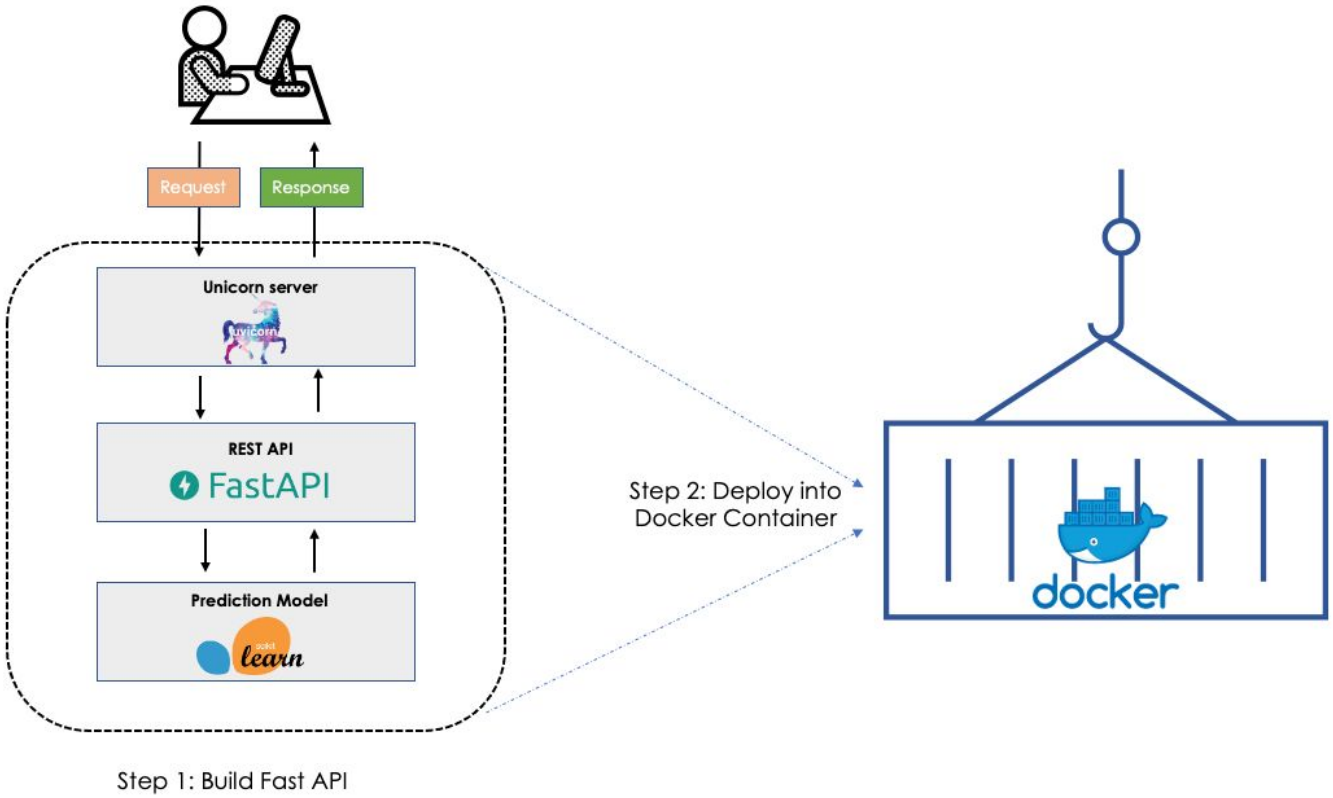
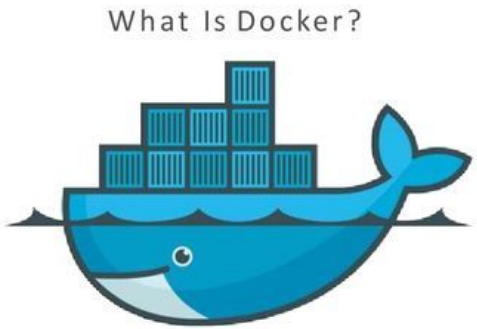
Docker


Request
Response
Unicorn server
REST API
FastAPI
Prediction Model
learn
Step 2: Deploy into
Docker Container
docker
Step 1: Build Fast API

Docker

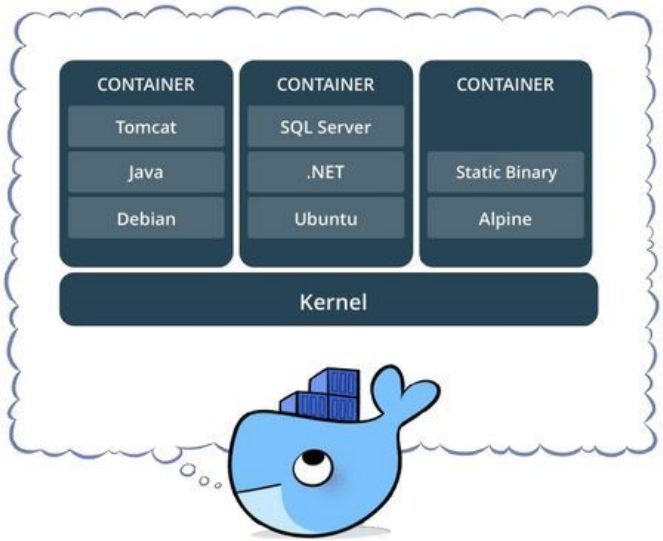


- Lightweight, open, secure platform
- Simplify building, shipping, running apps
- Runs natively on Linux or Windows Server
- Runs on Windows or Mac Development machines (with a virtual machine)
- Relies on "images" and "containers"

docker

Docker

What is a container?

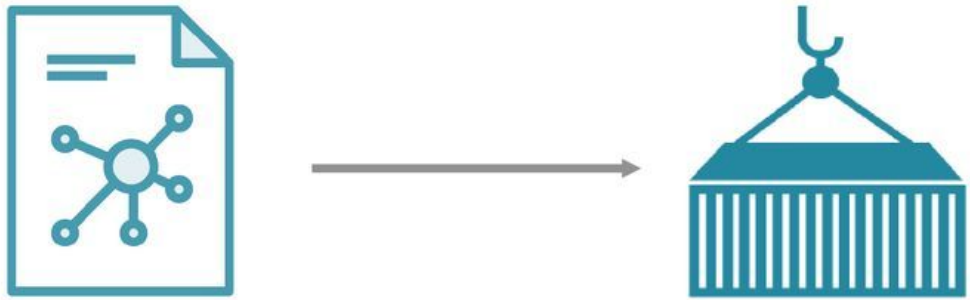


- Standardized packaging for software and dependencies
- Isolate apps from each other
- Share the same OS kernel
- Works for all major Linux distributions
- Containers native to Windows Server 2016

docker

Docker

The Role of Images and Containers



Docker Image

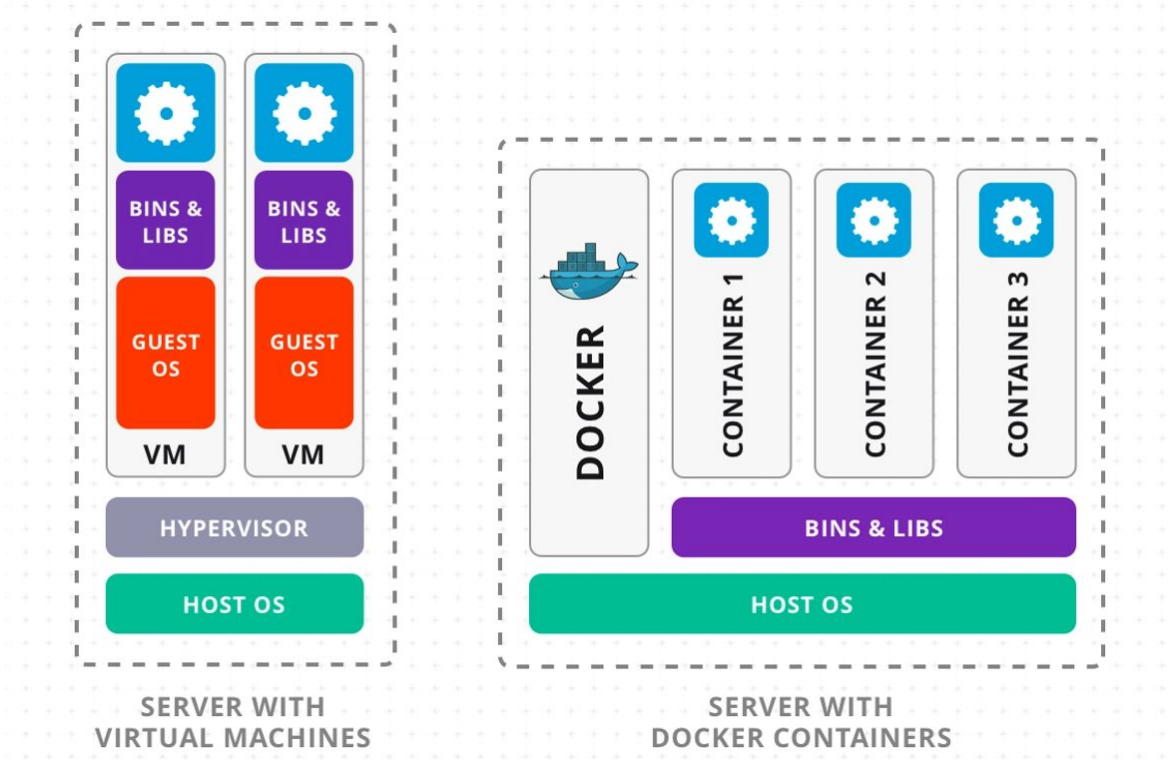
Example: Ubuntu with Node.js and Application Code

Docker Container

Created by using an image. Runs your application.

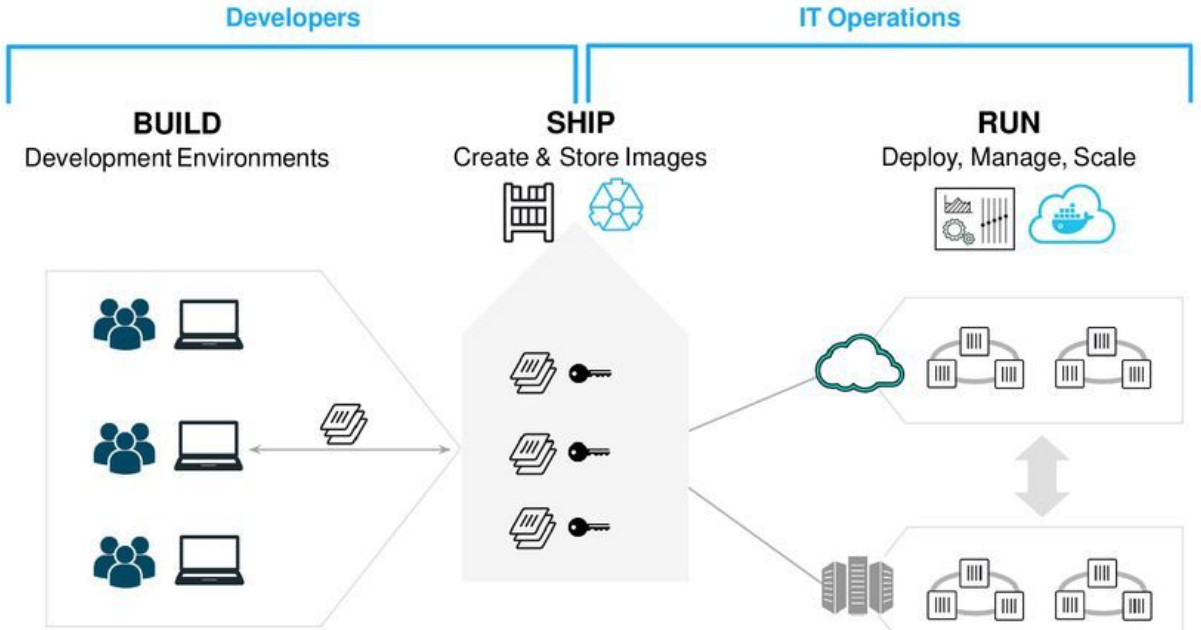
docker

Docker


BINS & LIBS
BINS & LIBS
GUEST OS
GUEST OS
VM
VM
HYPERVISOR
HOST OS
SERVER WITH VIRTUAL MACHINES
DOCKER
CONTAINER 1
CONTAINER 2
CONTAINER 3
BINS & LIBS
HOST OS
SERVER WITH DOCKER CONTAINERS

Docker

Using Docker: Build, Ship, Run Workflow



docker

Docker

Some Docker vocabulary

Docker Image

The basis of a Docker container. Represents a full application

Docker Container

The standard unit in which the application service resides and executes

Docker Engine

Creates, ships and runs Docker containers deployable on a physical or virtual, host locally, in a datacenter or cloud service provider

Registry Service (Docker Hub(Public) or Docker Trusted Registry(Private))

Cloud or server based storage and distribution service for your images

Docker

Basic Docker Commands

```
$ docker image pull node:latest

$ docker image ls
$ docker container run -d -p 5000:5000 --name node node:latest

$ docker container ps

$ docker container stop node(or <container id>)

$ docker container rm node (or <container id>)

$ docker image rmi (or <image id>)

$ docker build -t node:2.0 .

$ docker image push node:2.0

$ docker --help
```

Docker

Dockerfile

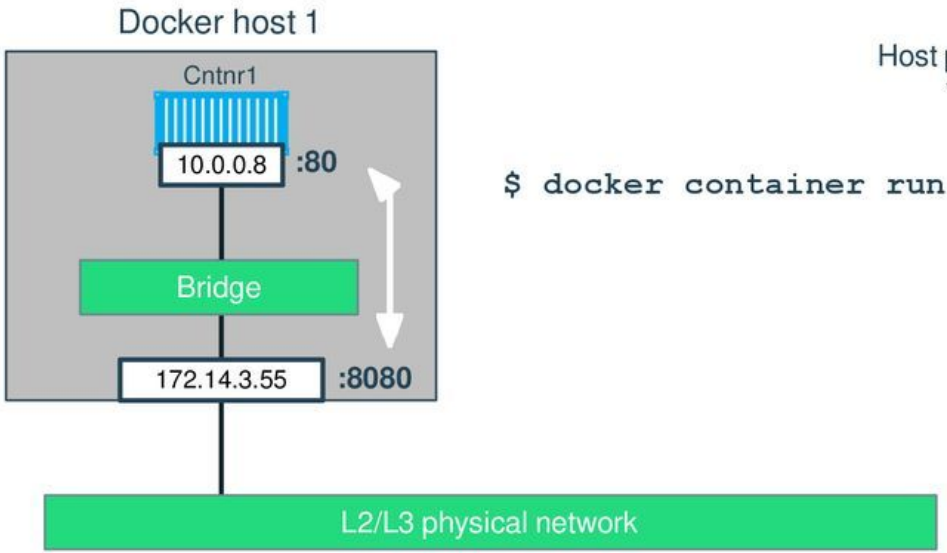
```
FROM node:12.16.3

WORKDIR /code

ENV PORT=80

# COPY package.json /code/package

RUN npm install

COPY . /code

# CMD ["node", "src/server.js"]
```

docker-compose.yml

```
services:
  app:
    build:
      context: .
    ports:
      - "5000"


  nginx:
    image: nginx:latest
    volumes:
      - ./nginx.conf:/etc/nginx/nginx.conf:ro

    depends_on:
      - app

    ports:
      - "80:80"

```

Docker

Docker Bridge Networking and Port Mapping



Host port Container port

```
$ docker container run -p 8080:80 ...
```


docker