

# Docker Containers and Deploying .NET/ASP.NET Core App in Containers

Ehsan Eskandari

IT Consultant

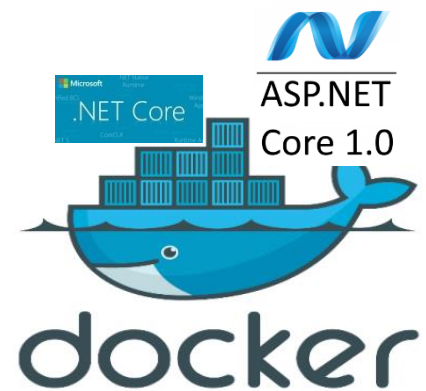
@EhsanEskandarim

#Docker, #.NETCORE, #ASPNETCORE, #WindowsContainer

Toronto .NET Meetup

Microsoft Downtown Office

2017/02/22



# Agenda

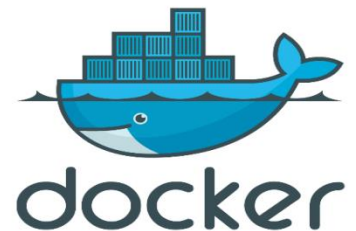
- Docker Containers
- Docker Architecture
- Docker Benefits
- ASP.NET Core
- Sharing Files
- Demo

# Agenda

- Docker Containers
- Docker Architecture
- Docker Benefits
- ASP.NET Core
- Sharing Files
- Demo

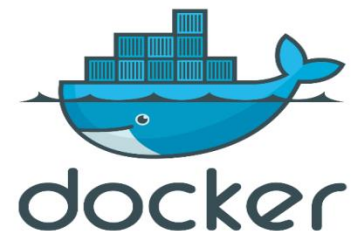
# What is Docker?

- Docker containers wrap a piece of software in a complete filesystem that contains everything needed to run: code, runtime, system tools, system libraries – anything that can be installed on a server. This guarantees that the software will always run the same, regardless of its environment.



# What is Docker?

- Lightweight, open source, secure platform
- Simplifies building, shipping, running apps
- Shipping container system for delivering code
- Runs natively on Linux and Windows Server 2016 or higher
- Relies on Images and Containers



# What is Docker?



# What is Docker?

Obligatory norms for European containers since 1 July 1933

Category	length [m]	width [m]	high [m]	Total mass [tons]
Heavy types				
Close type 62	3.25	2.15	2.20	5
Close type 42	2.15	2.15	2.20	
Open type 61	3.25	2.15	1.10	
Open type 41	2.15	2.15	1.10	
Light Type				
Close type 22	2.15	1.05	2.20	2.5
Close type 201	2.15	1.05	1.10	
Open type 21	2.15	1.05	1.10	

In April 1935 BIC established second standard for European containers:<sup>[8]</sup>

Obligatory norms for European containers since 1 April 1935

Category	Length [m]	Width [m]	High [m]	Total mass [tons]
Heavy types				
Close 62	3.25	2.15	2.550	5
Close 42	2.15	2.15	2.550	
Open 61	3.25	2.15	1.125	
Open 41	2.15	2.15	1.125	
Light Type				
Close 32	1.50	2.15	2.550	2.5
Close 22	1.05	2.15	2.550	





# What is Docker?

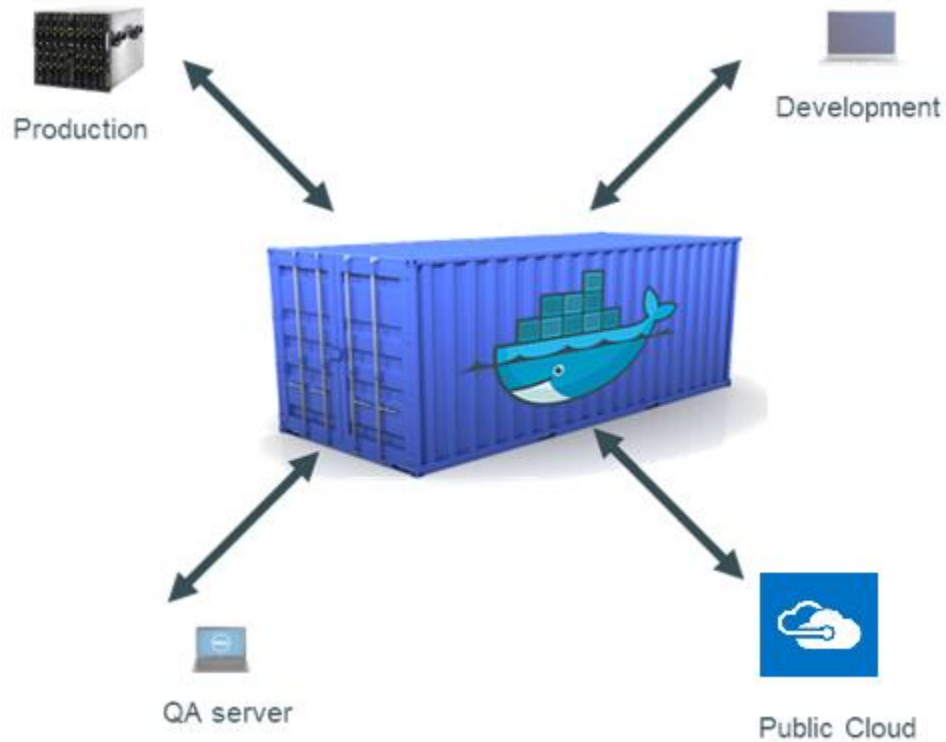
Efficiency in

- Shipping Speed
- Shipping Costs



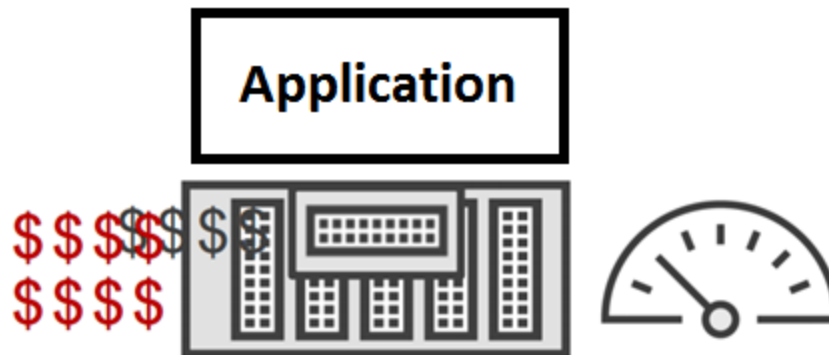


# What is Docker?



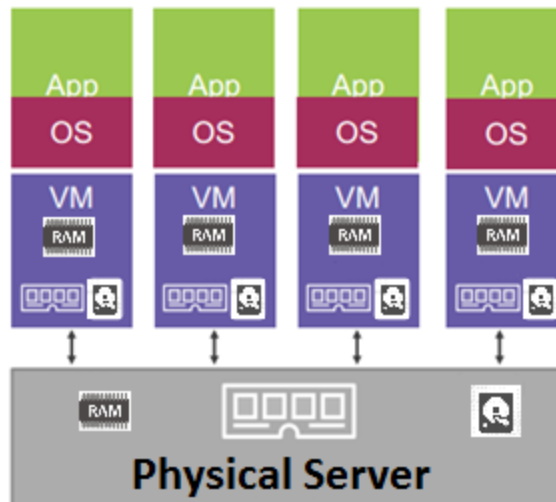
# Servers in Old days

- One application on one physical server.
  - Procurement
  - Capex and Opex costs
  - Costs: Hardware, Power, Administration, etc.
  - Massively Over-Powered Servers with 10% CPU consumption



# Virtualization and Hypervisor

- Technology that enables us having Same Physical Server and squeeze so much more
- Instead of one application on one physical server, now multiple application on one physical server.



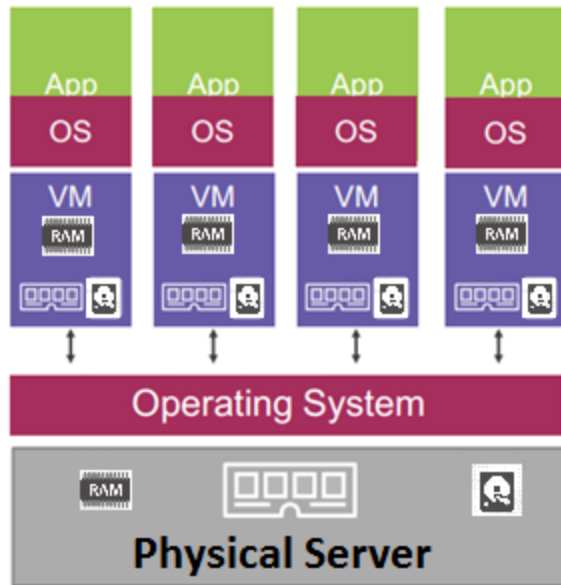
# Virtualization Issues

- One physical server
- $n$  virtual machines for  $n$  applications
- These vm are slices of physical hardware, e.g. allocating 25% of cpu power, 25% memory, 25% disk space
- These vm are slices of real resources in the physical server
- Each of these vms needs a dedicated OS and each of which uses cpu, memory, and disk
- So far it is just OS and there is no application yet
- Each OS needs license \$\$\$\$
- Each OS needs admin e.g. patching, anti-virus

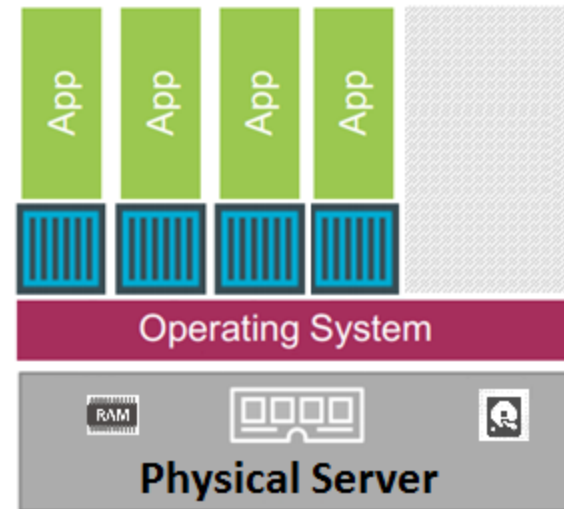
# Docker containers

- **Instead of installing Hypervisor and virtual machines, we install only one OS and on top that we create 4 containers. Then applications are running inside each containers.**
- **Containers are much smaller and more efficient than VMs**

# Containers vs Virtualization



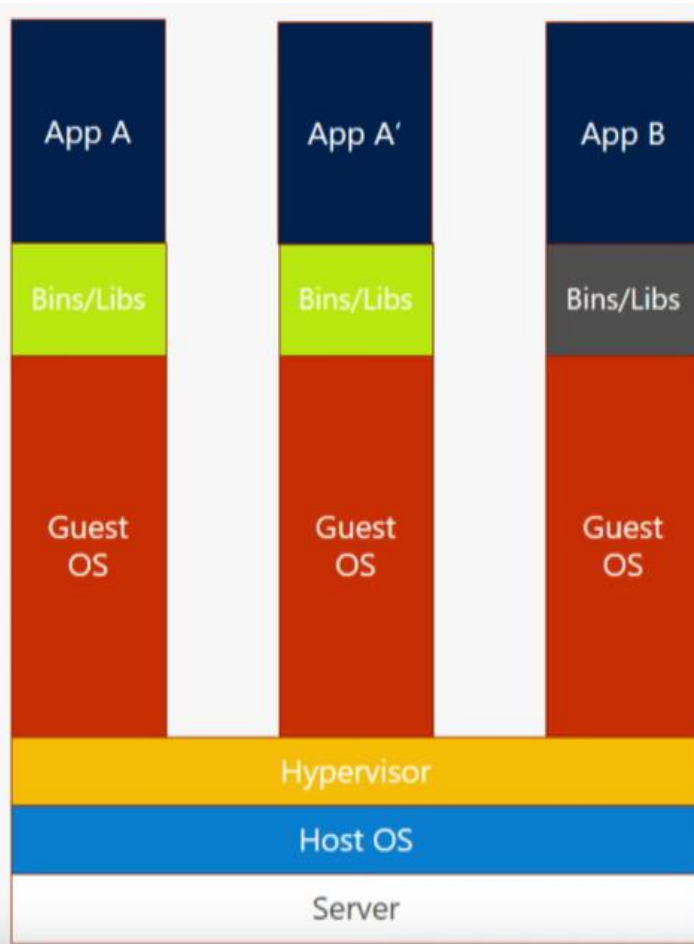
Hypervisor  
Architecture



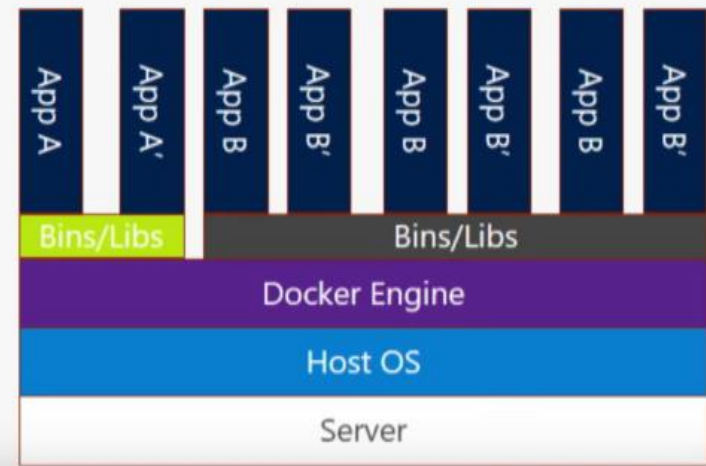
Container  
Architecture



# Containers vs Virtualization



Containers are isolated, but share OS and, where appropriate, bins/libraries



# Windows Containers

Windows Server 2016 or higher built in

Development machine:

Docker for windows and windows 10 Professional or enterprise

Enable Hyper-V and containers

## Windows Server Container

## Hyper-V Container

Isolation	Process	VM
Windows Server 2016	Yes	Yes
Windows 10	No	Yes

# Windows Containers

## Docker on windows development ENV

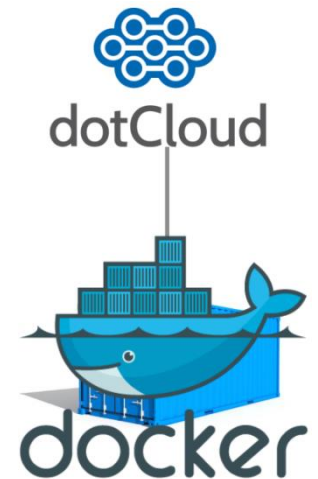
- 1.Install windows 10 Pro or Enterprise
- 2.Enable virtualization in bios level
- 3.Install hyper-v
- 4.Install docker for windows

# Docker

- an [open-source](#) project that automates the deployment of applications inside software containers
- Docker is the containers What Vmwares is to the Hypervisor
- Docker Inc.(the company)
- Docker project: container engine plus Docker tools e.g. Docker client, Docker compose...
- [github Repository](#)

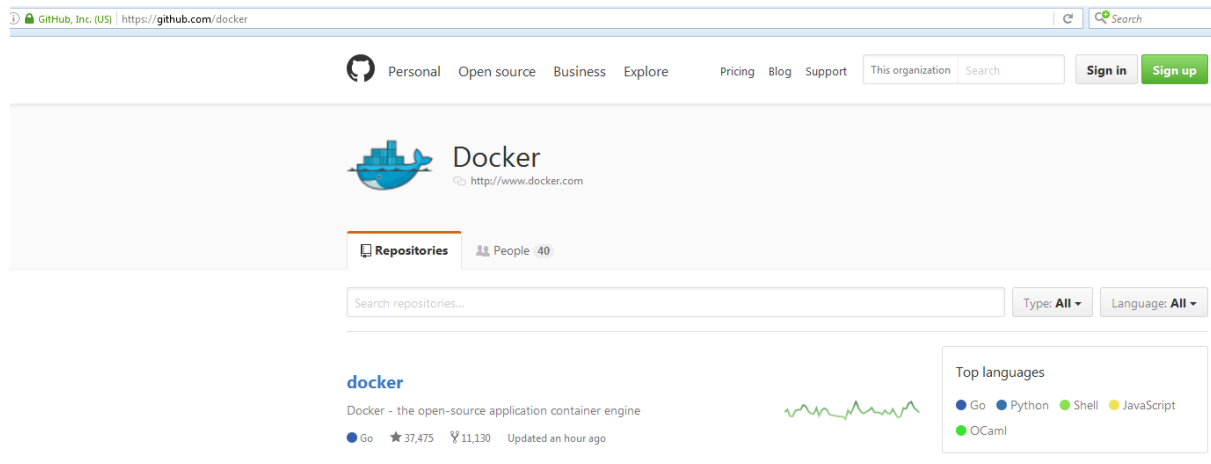
# Docker Inc.

- The original author and primary sponsor of the Docker open source project.
- Tech Starter based on SF
- Originally it was dotCloud (platform as a service) and the idea behind that business was to offer the developer platform on top of amazon web service
- [Docker Inc. Website](#)
- Docker Inc has over 120 employees



# Docker Project

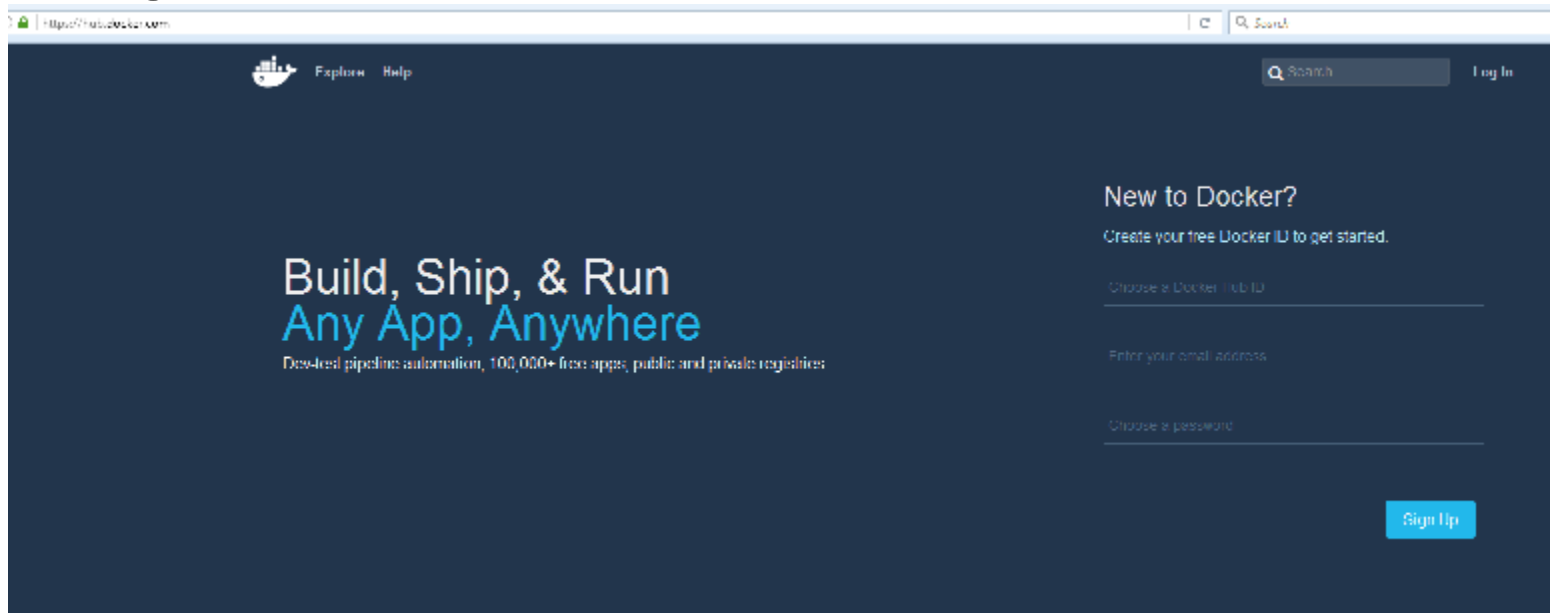
- The Docker project is not absolutely the Docker Inc.
- It belongs to the community.
- Open source under Apache License 2.0
- Core Docker component written in GO language
- [github Repository](https://github.com/docker/docker)





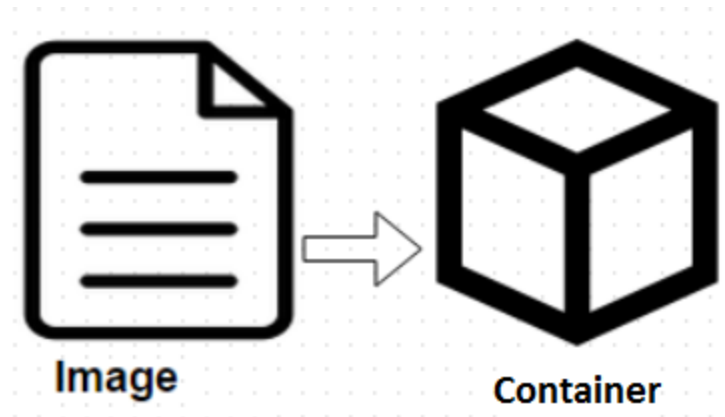
# Docker Hub

- a cloud-based registry service which allows you to link to code repositories, build your images and test them, stores manually pushed images.
- The public Docker registry(Hub.Docker.com).
- Searchable.
- Private Registries: can be hosted in Docker, Azure container registry, AWS, Google



# Docker Image

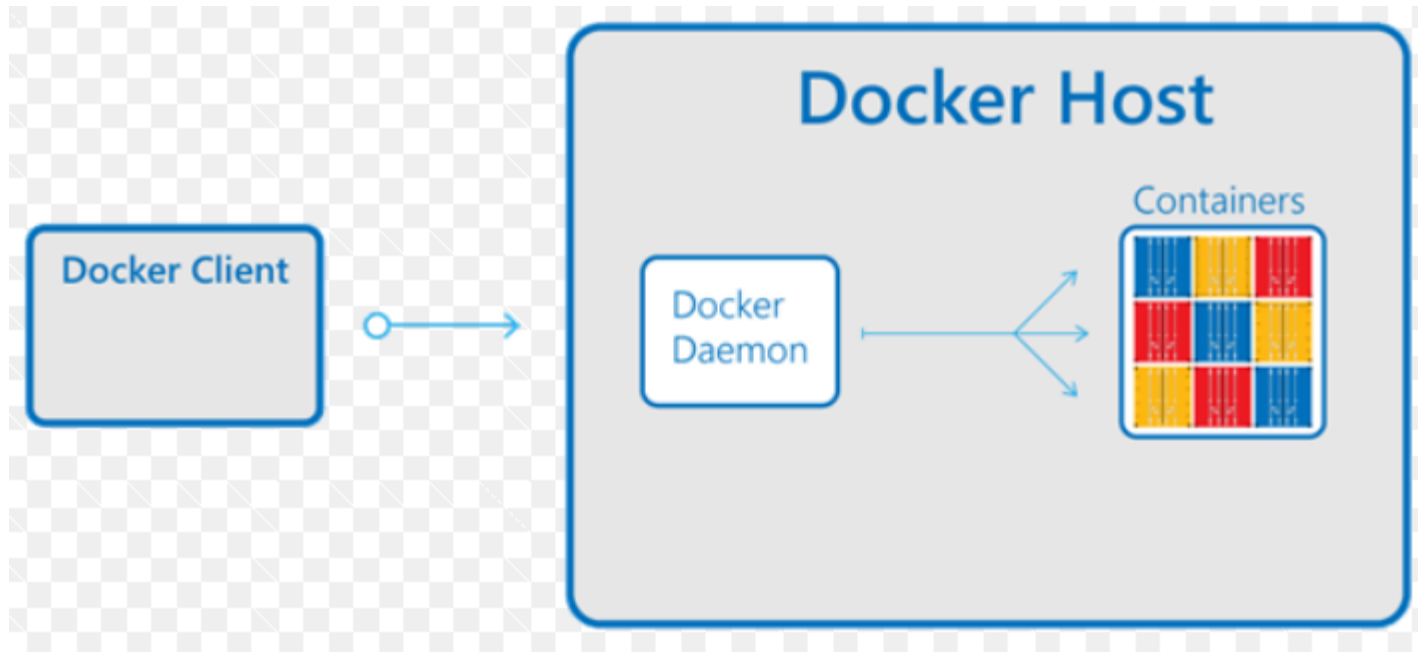
- An image is an immutable file that's essentially a snapshot of a container.
- Images are created with the build command
- They'll produce a container when started with run command.
- Images are stored in a Docker registry such as [registry.hub.docker.com](https://registry.hub.docker.com).
- Some related Docker client commands
  - `docker images`
  - `docker run image-name`
  - `docker stop image-id`
  - `docker start image-id`
- Image: is a virtual machine template



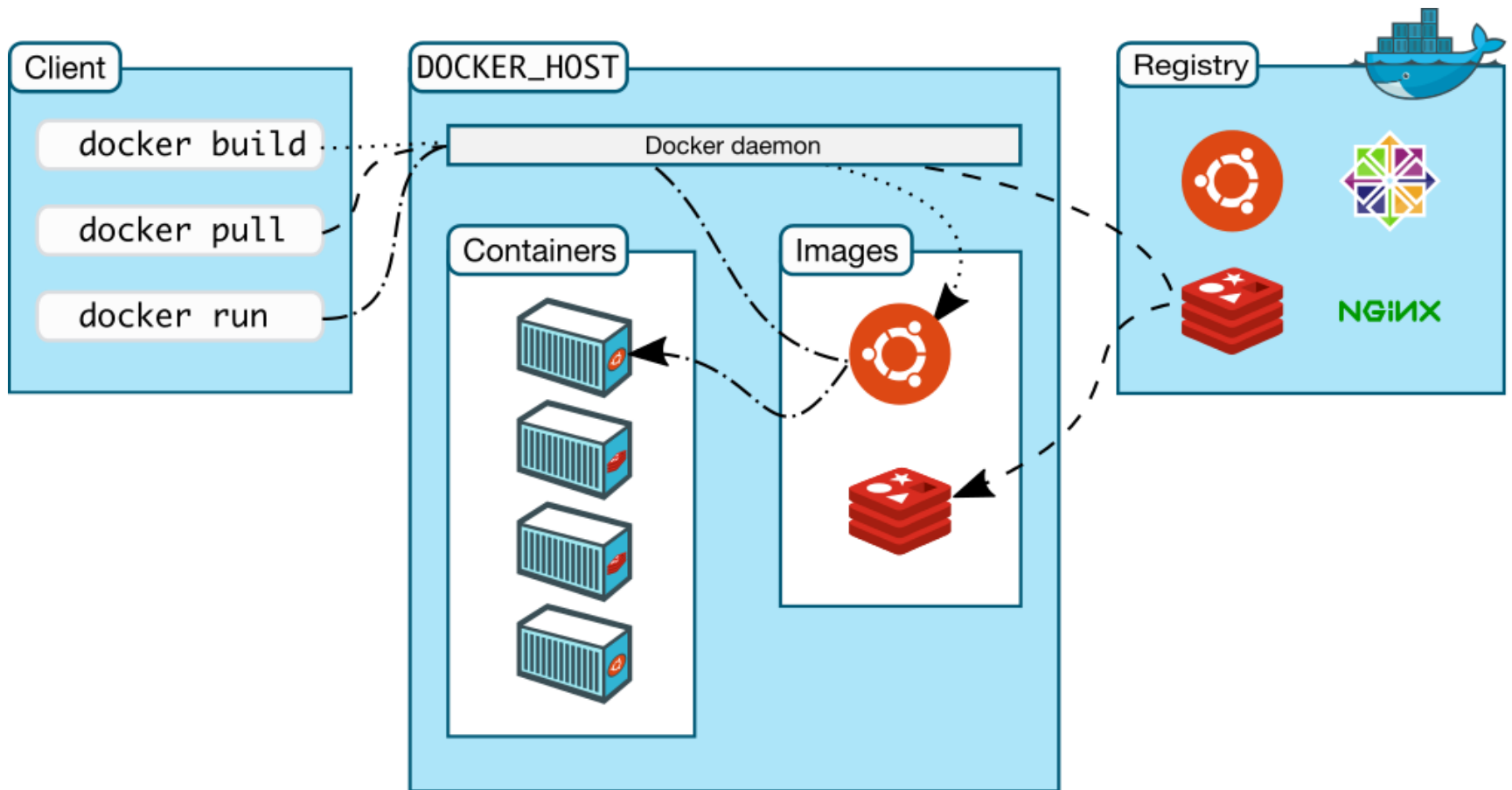
# Agenda

- Docker Containers
- **Docker Architecture**
- Docker Benefits
- ASP.NET Core
- Sharing Files
- Demo

# Docker Architecture



# Docker Architecture



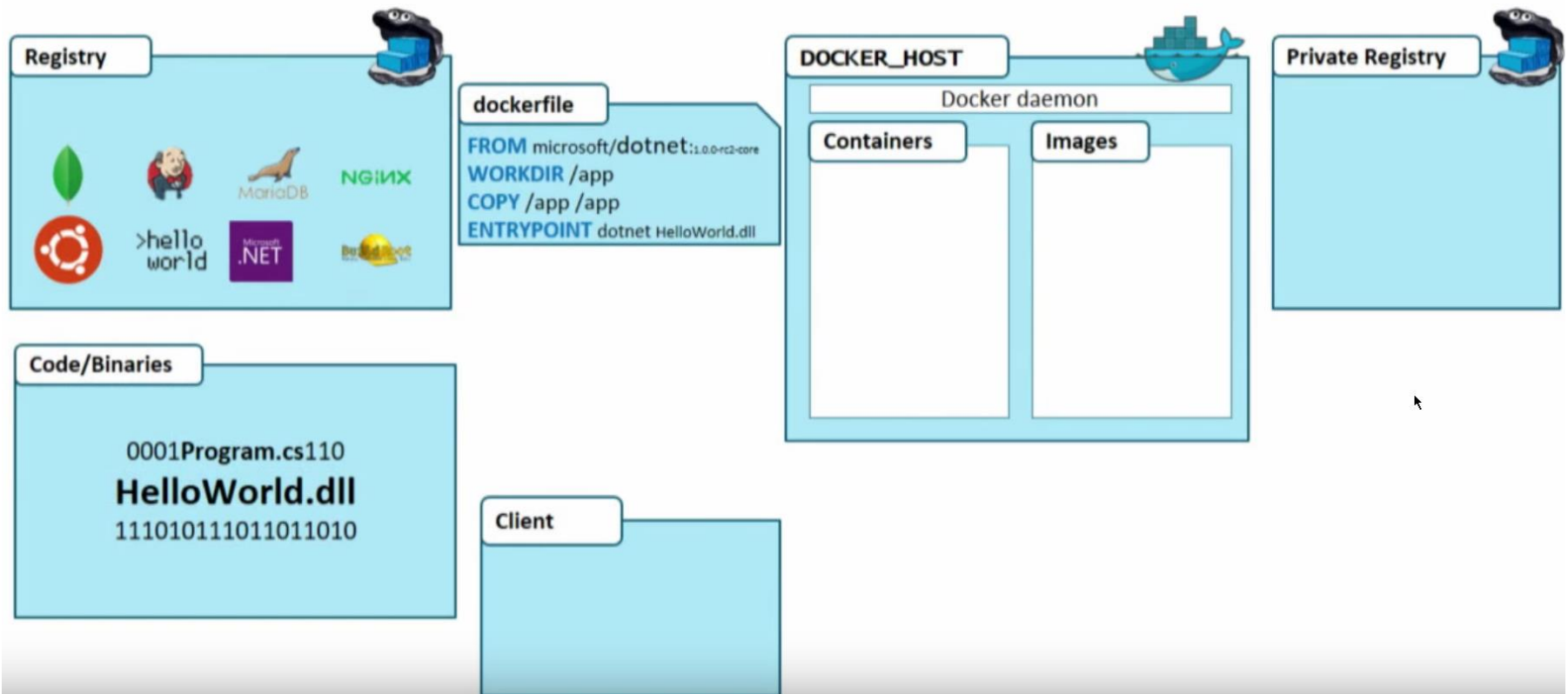
# Docker Architecture

- Install Ubuntu
- In my machine the host OS is Ubuntu but if you have a windows machine you can install Ubuntu or any other Linux distro using Virtual box.
- Sudo su
- apt-get install -y docker.io
- Docker version => gives both client and server version
- Docker info => give info about containers status and images
- Docker pull IMAGE\_NAME => pull image locally
- Docker images
- Docker run IMAGE\_NAME
- Docker ps => list running containers
- Docker ps -a => list all containers
- Docker stop IMAGE\_NAME => stop container
- Docker rm IMAGE\_NAME => remove stopped container
- Docker rmi IMAGE\_NAME => remove image from docker host



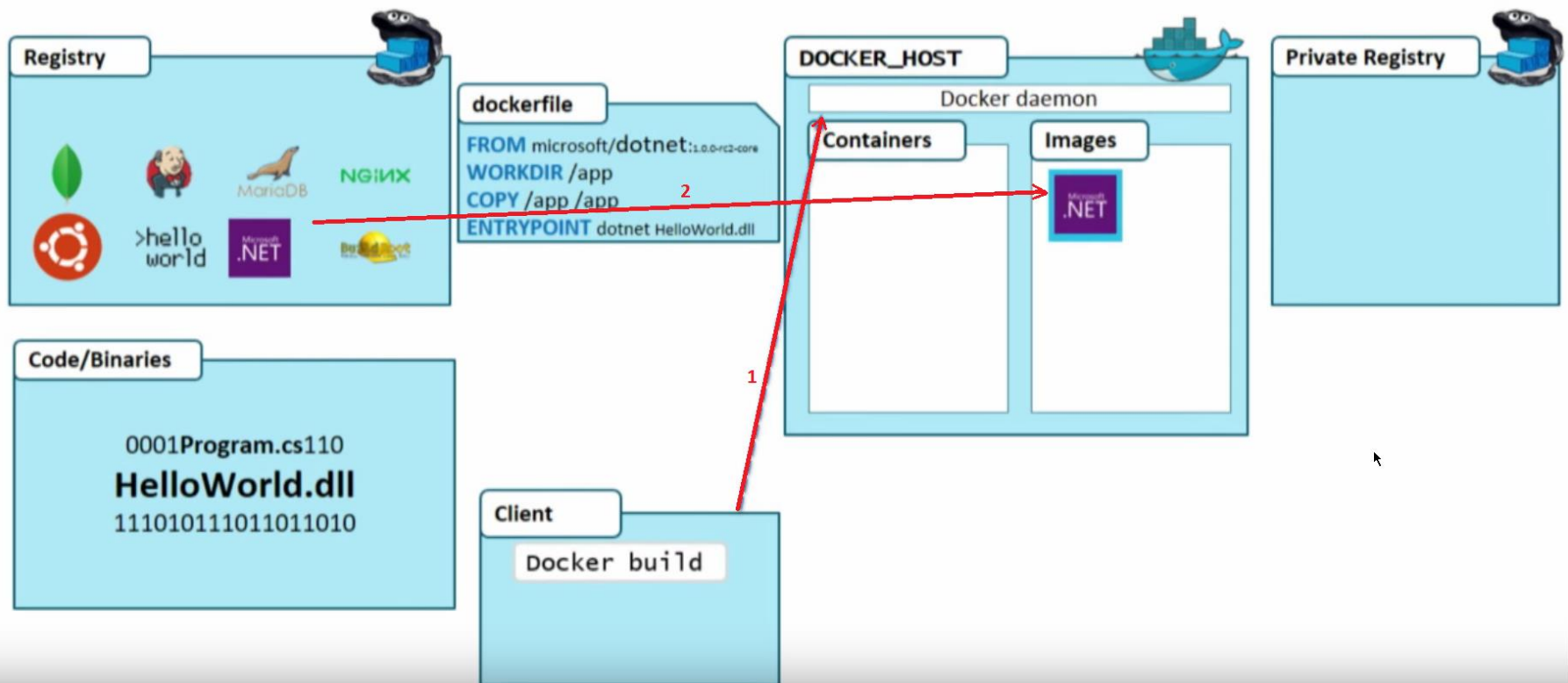
# Docker Architecture

## Docker build



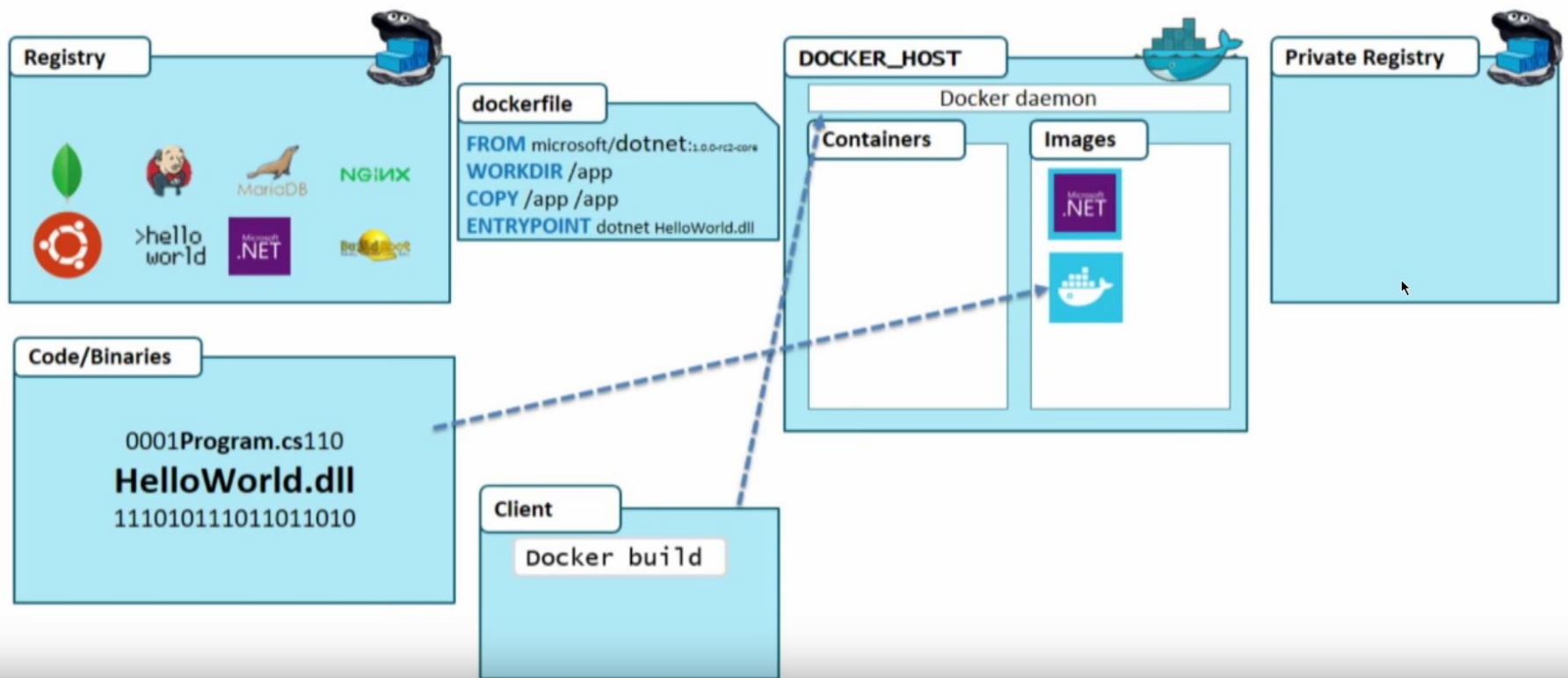
# Docker Architecture

## Docker build



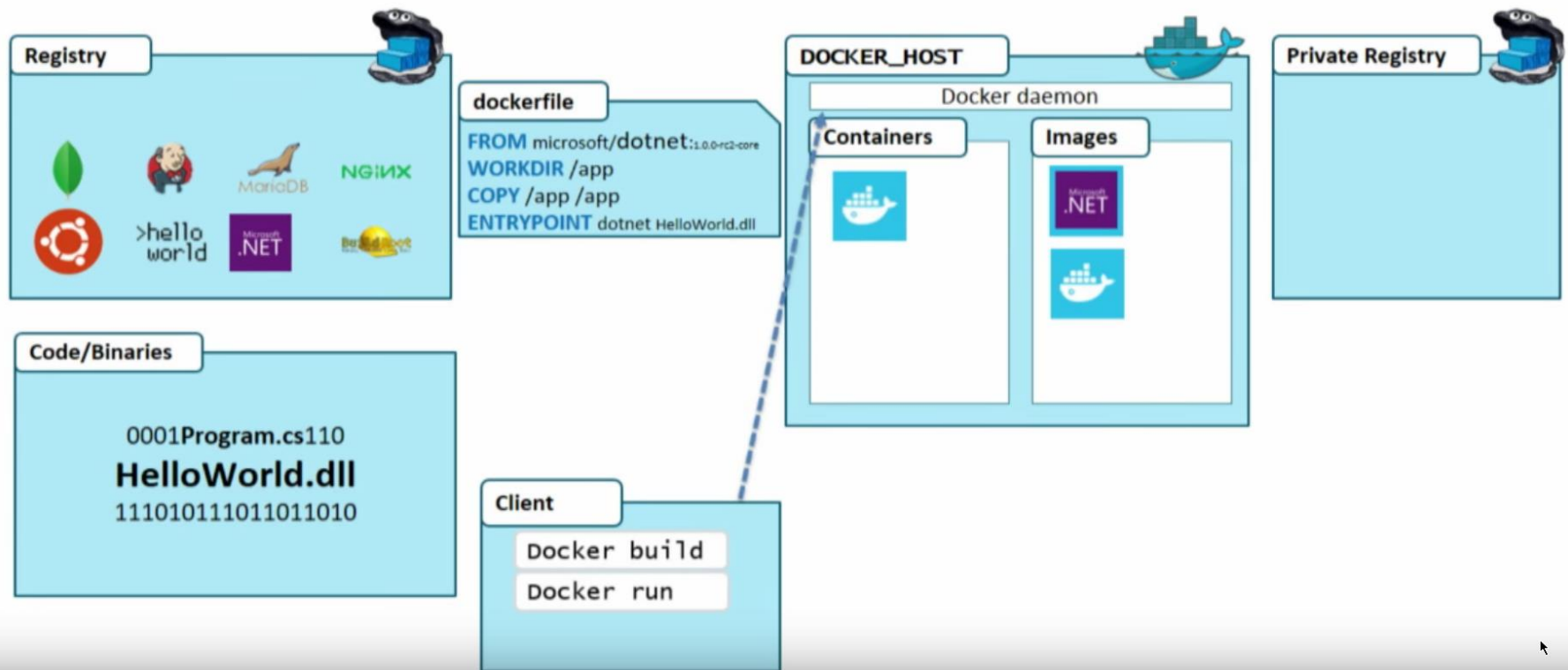
# Docker Architecture

## Docker build



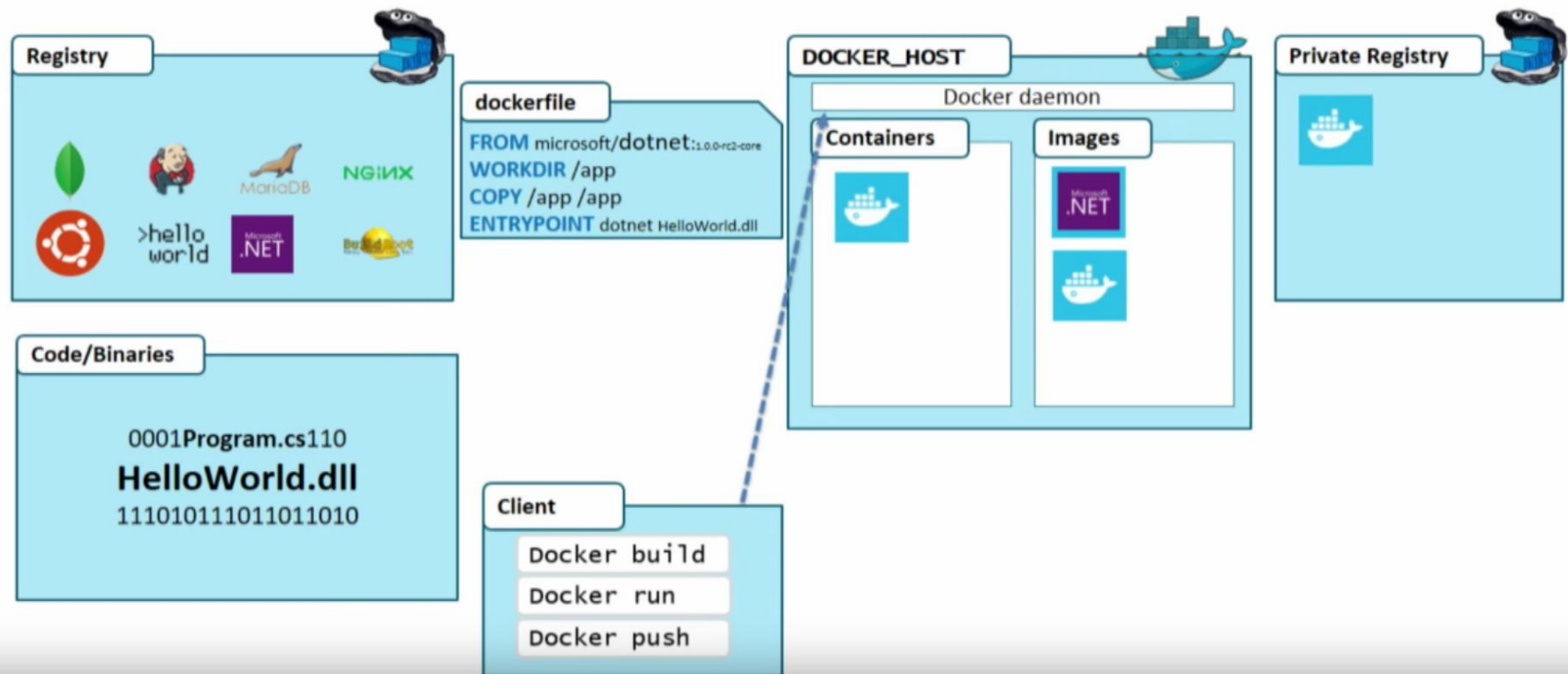
# Docker Architecture

## Docker build



# Docker Architecture

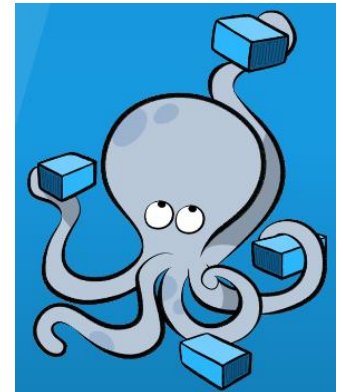
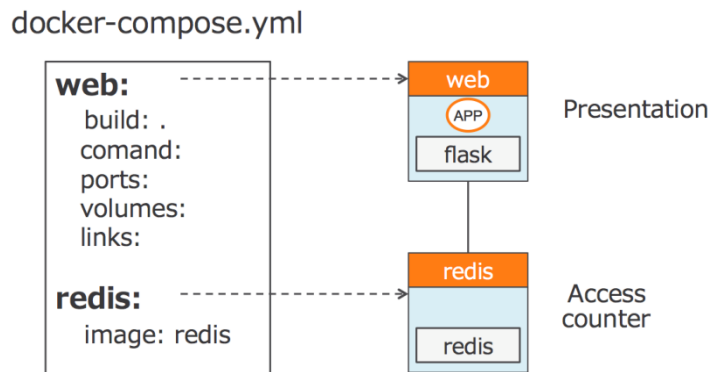
## Docker build



# Docker Architecture

## Docker Compose:

- a tool for defining and running multi-container Docker applications.
- Compose file to configure your application's services. Then, using a single command, you create and start all the services from your configuration.
- API above docker run
- Docker-compose.yml





# Agenda

- Docker Containers
- Docker Architecture
- **Docker Benefits**
- ASP.NET Core
- Sharing Files
- Demo

# Docker Benefits

- Set up DEV environment very quickly
- No More App Conflict
- Consistent Environment
- Faster Software Shipment

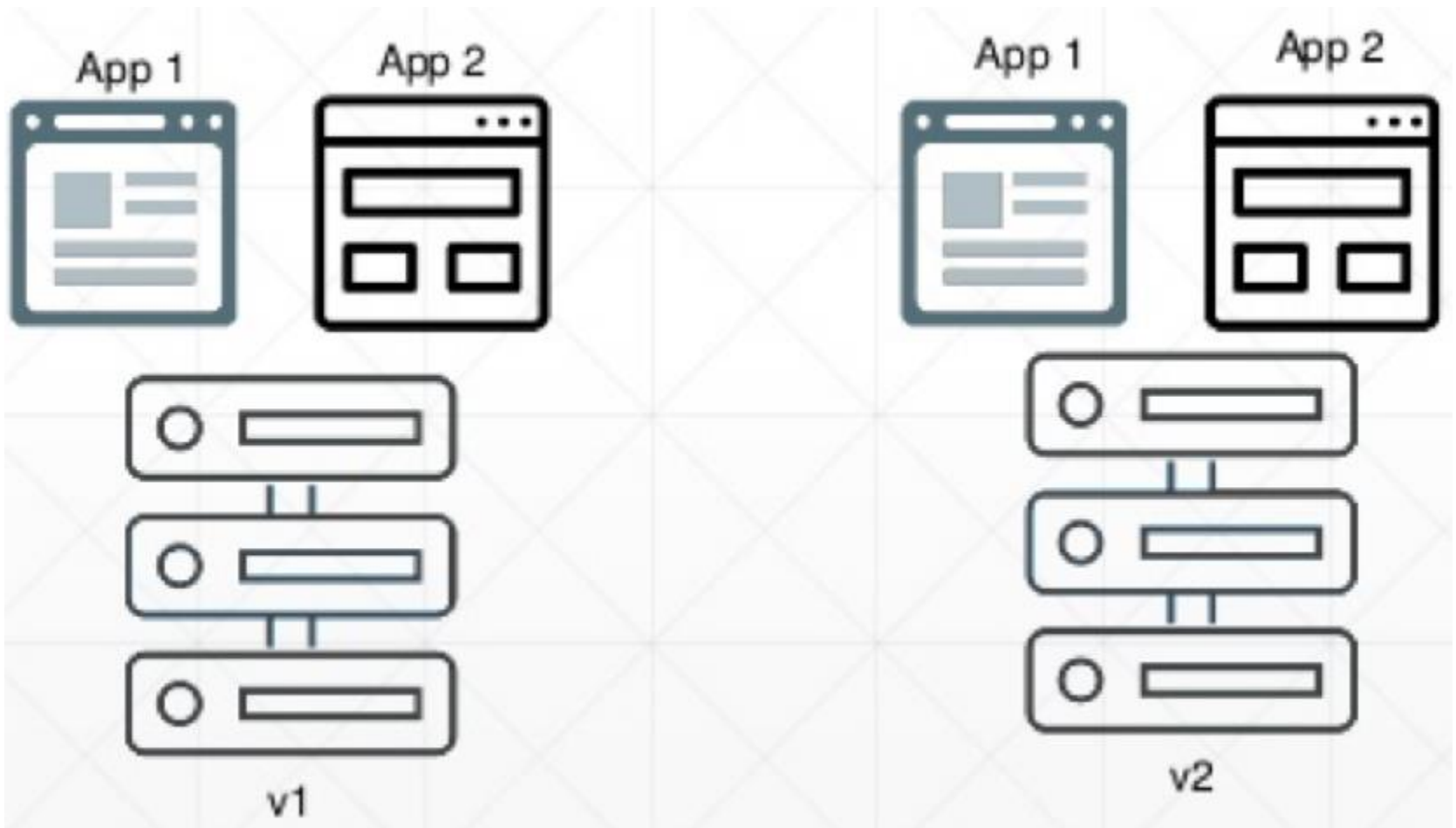
# Docker Benefits

- Set up DEV environment very quickly



# Docker Benefits

- No More App Conflict



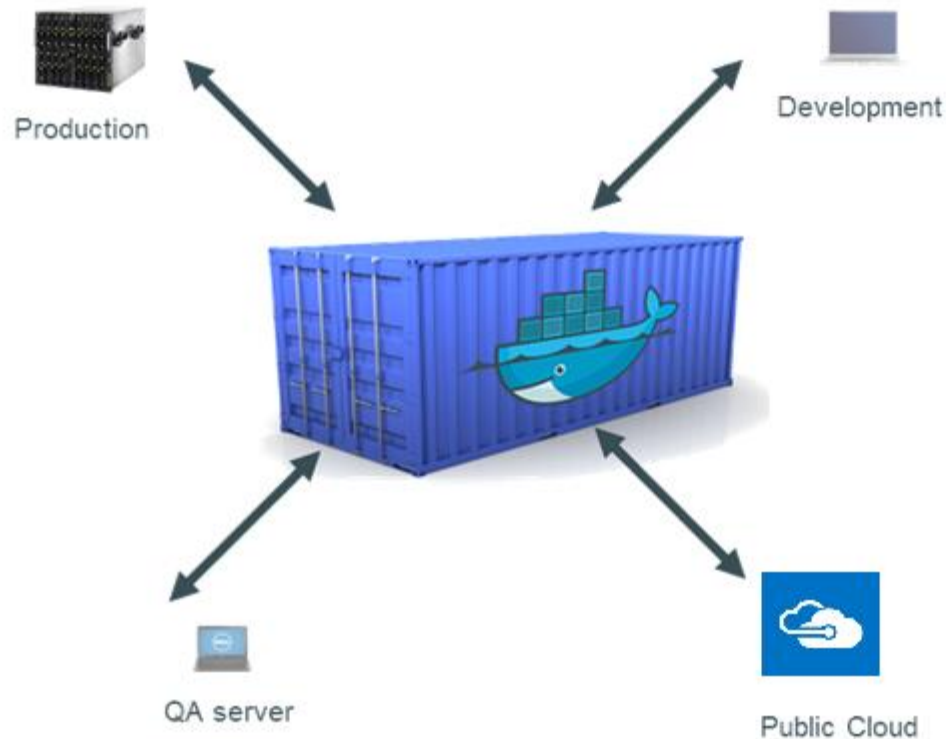
# Docker Benefits

- Consistent Environment



# Docker Benefits

- Faster Software Shipment



# Agenda

- Docker Containers
- Docker Architecture
- Docker Benefits
- **ASP.NET Core**
- Sharing Files
- Demo

# ASP.NET Core

- Cross Platform .NET
  - Mono
  - .NET Core





# ASP.NET Core

- Mono
  - Mono is an open source implementation of Microsoft's .NET Framework
  - Sponsored by Microsoft
  - Joined the .NET Foundation

Cross platform, open source .NET framework



# ASP.NET Core

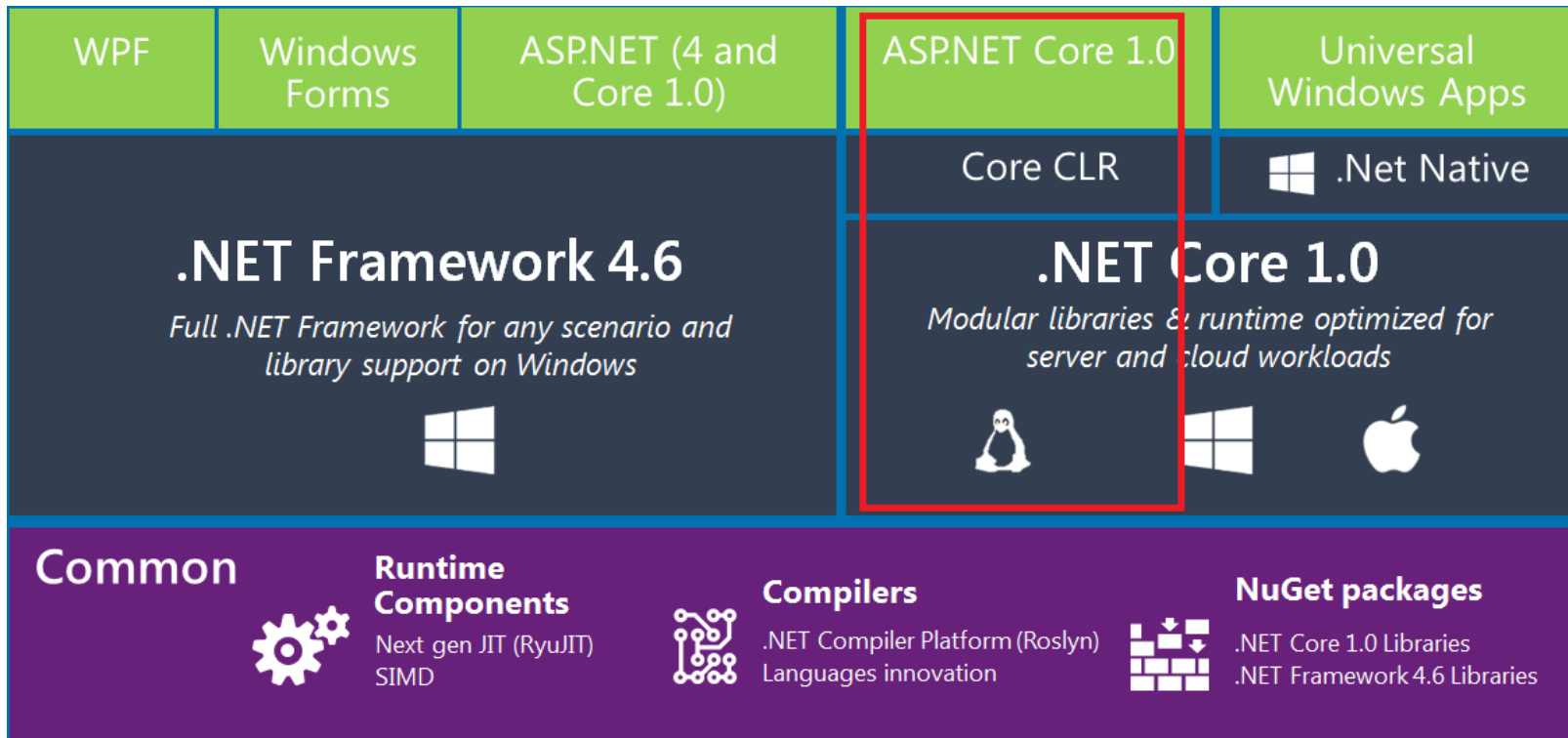
- .NET Core
  - Lightweight open source framework
  - Runs on Windows, Linux, OSX
  - Less features than old .NET Framework



# ASP.NET Core

- ASP.NET Core is a lean and composable framework for building web and cloud applications.
- Fully open source and available on [GitHub](#)
- Formerly ASP.NET 5
- Runs on .NET Framework and .NET Core
- Available on Windows, Mac, and Linux

# ASP.NET Core



# ASP.NET Core

- **ASP.NET Core and Docker**
  - microsoft/dotnet image
  - Official base image for .NET Core applications
  - Based on Debian
  - Uses the “stable” version of .NET Core

# ASP.NET Core

microsoft/aspnet - Docker ...

https://hub.docker.com/r/microsoft/aspnet/

Search

Sign up Log In

PUBLIC | AUTOMATED BUILD

microsoft/aspnet ☆

Last pushed: 2 days ago

Repo Info

Tags

Dockerfile

Build Details

Short Description

ASP.NET is an open source server-side Web application framework

Full Description

### Supported tags and respective Dockerfile links

- 4.6.2-windowsservercore-10.0.14393.321, 4.6.2-windowsservercore, 4.6.2, latest (4.6.2/Dockerfile)
- 3.5 (3.5/Dockerfile)

These images are updated via [pull requests to the microsoft/aspnet-docker GitHub repo](#).

docker pulls 1M

docker stars 524

### What is ASP.NET


ASP.NET is a high productivity framework for building Web Applications using Web Forms, MVC, Web API and SignalR.

This repository contains Dockerfile definitions for ASP.NET Docker images. These images use the [IIS image](#) as their base.


Docker Pull Command

`docker pull microsoft/aspnet`

Owner

 microsoft

Source Repository

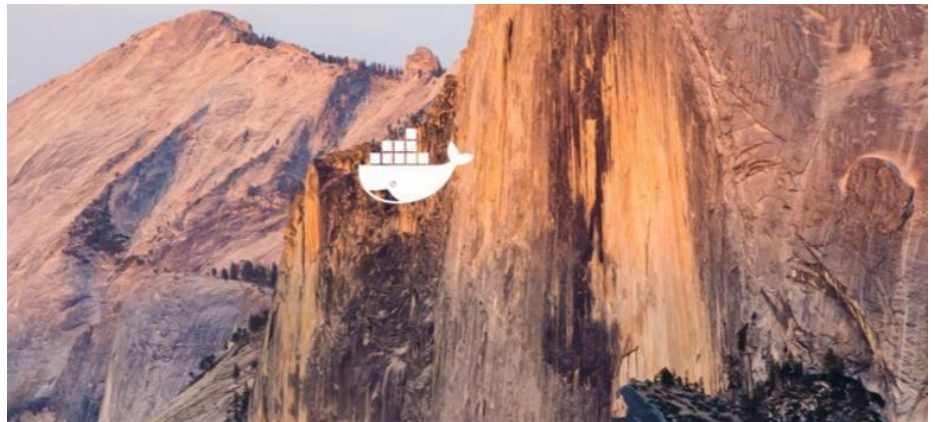
 [aspnet/aspnet-docker](#)

# Agenda

- Docker Containers
- Docker Architecture
- Docker Benefits
- ASP.NET Core
- **Sharing Files**
- Demo

# Sharing Files

- Docker images are immutable
- To persist data, we need an external storage in scenarios like log, database files



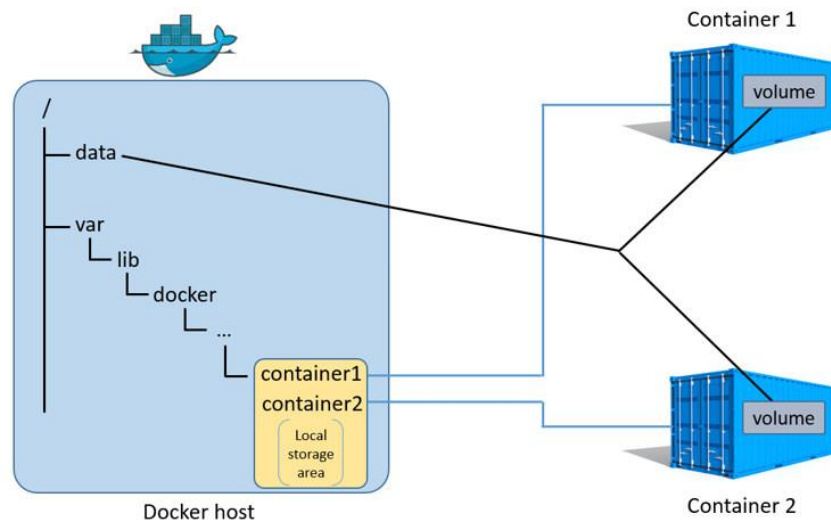


# Sharing Files

- Image Level: Read only, good for binaries and source codes
- Container Level: goes away if container removed
- Data Volume: good for any data to persist e.g. logs, database files etc

# Volumes

- Data Volume: is a special type of directory in containers. Similar to a network mapped drive.
- Separate from image and containers. Any changes to the image/container won't impact volume.
- They can be shared among other containers e.g. one source code and running multiple instance of the code on different containers.
- Persisted to the Docker host.
- `docker run -p 8080:5000 -v $(pwd):/app microsoft/aspnet`



# Agenda

- Docker Containers
- Docker Architecture
- Docker Benefits
- ASP.NET Core
- Sharing Files
- **Demo**





1. Docker client commands
2. Full .NET Core build life cycle
3. Deploy ASP.NET Core on docker
4. Windows Containers and SQL SERVER

# 1.Docker client commands

- docker version + switch between windows/linux
- docker images
- docker pull
- docker ps
- docker run
- docker rm
- docker rmi
- docker search microsoft --no-trunc
- docker login
- docker build
- docker push
- docker inspect

# 2. Full .NET Core build life cycle

Now in cmd cd in a new folder and run: dotnet new

**dotnet new**

Then this will create a new project

In this new project we need to add a dockerfile

In the docker file, usually we need two types of images, one for development and another one more efficient, fast, small for production.

So 'From' would say from this images. Then we would say inside the container set the working directory to /app

Then copy from /app folder from host machine to the app folder of the container

Then we would say what is the entry point when the container starts. This is not like windows server and have tones of services running.

So I would say ENTRYPOINT dotnet ReadFromFileDocker.dll

**FROM microsoft/dotnet:latest**

**WORKDIR /app**

**COPY /app /app**

**ENTRYPOINT dotnet ReadFromFileDocker.dll**

Now we need to run 'dotnet restore' in powershell to restore all required packages

**dotnet restore**

Then we would also do 'dotnet build -c release'

**dotnet build -c release**

Now notice that we have a bin folder. Now, I would say 'dotnet publish -c release -o app'

**dotnet publish -c release -o app**

And it will create the app directory in vs code. Now that I have my app build, I need to build docker build

I am tagging it with -t and also add a dot at the end to look for dockerfile in the current context:

**Docker build -t readfromfiledocker .**

Now we have built our images and if we do a docker images we will have this images in the list.

**docker images**

Now let's run this images by running docker run test20170218

**docker run readfromfiledocker**

Okay, so we are in a windows machine, we just compiled this .net application. Then we have built an image. Then we ran it inside a linux container on my windows machine. In addition to that we can put all other required dependencies for our application in the docker file and have the environment ready to run our application.

Now let's tweak this app a little bit and see. For example let's put the app code in a loop and retry it for 30 times. After code change we need to do dotnet publish -c release -o app

To rebuild it for us and then we need to run=>

**docker build -t readfromfiledocker:version2 .**

Now for push:

Docker tag readfromfiledocker eskandari/toronto\_dotnet\_meetup\_sample

Docker login --username=eskandari

Docker push eskandari/toronto\_dotnet\_meetup\_sample

Now, let's make a change to our code, and then save

```
dotnet publish -c release -o app
```

```
Dotnet run
```

```
Docker build -t readfromfiledocker .
```

```
Docker tag readfromfiledocker eskandari/toronto_dotnet_meetup_sample
```

```
Docker push eskandari/toronto_dotnet_meetup_sample
```

Then on linux machine:

```
Docker pull eskandari/toronto_dotnet_meetup_sample
```

```
Docker run eskandari/toronto_dotnet_meetup_sample
```



# 3. Deploy ASP.NET Core on docker

Use linux machine:

```
sudo docker run -i -t -p 8080:5000 -v $(pwd):/app -w "/app" microsoft/aspnet:1.0.0-rc1-update1-coreclr /bin/bash  
-then dnu restore  
-then dnx web
```

Then using visual studio update a view file and see the result in the browser.

Then go to AWS UBUNTU Server => 35.160.239.107 and then run the fiddler and show the kestrel web server

Then using WinSCP update a view on the cloud and then show the result in the browser.

# 4. Windows Containers and SQL SERVER

first make sure that there is no other SQL SERVER container by `docker ps -a`

if so use `docker rm CONTAINER_ID` to remove it

**`docker run -it -p 1433:1433 -e sa_password=P@ssw0rd -e ACCEPT_EULA=Y microsoft/mssql-server-windows-express`**

Then run `docker inspect` and get the ip address and then login using sql server management studio

OR Update-Database from package manager console then after running go to <http://localhost:7223/pie/list>

Login to azure and run a vm

In azure:

`Docker run -it microsoft/dotnet:nanoserver powershell`

Inside it call `=> get-process`

Then open a new powershell in the host and `get-process` and then find the process id of the container in the host and you will find it. That shows this is separate in process

But if you do the same in windows 10, you can not find it.

## 5. Running ASP.NET Core application and SQL SERVER on a windows container with docker compose.

