How to make the best use of Live Sessions

- Please login on time
- Please do a check on your network connection and audio before the class to have a smooth session
- All participants will be on mute, by default. You will be unmuted when requested or as needed
- Please use the “Questions” panel on your webinar tool to interact with the instructor at any point during the class
- Ask and answer questions to make your learning interactive
- Please have the support phone number (US : 1855 818 0063 (toll free), India : +91 90191 17772) and raise tickets from LMS in case of any issues with the tool
- Most often logging off or rejoining will help solve the tool related issues

edureka!

COURSE OUTLINE

Module 01

Introduction to Kubernetes

Kubernetes Architecture

Deploy app to Kubernetes Cluster

Expose App, Scale App And Update App

Managing State with Deployments

Federations, Auditing and Debugging Kubernetes, Security best practices

edureka!

Introduction to Kubernetes

Objectives

After completing this module, you should be able to understand:

- Docker - Essentials & a Short Recap
- Introduction to YAML
- What is Virtualization?
- What is Containerization?
- Virtualization v/s Containerization
- Kubernetes - A container-centric platform
- What Kubernetes is not

Virtualization

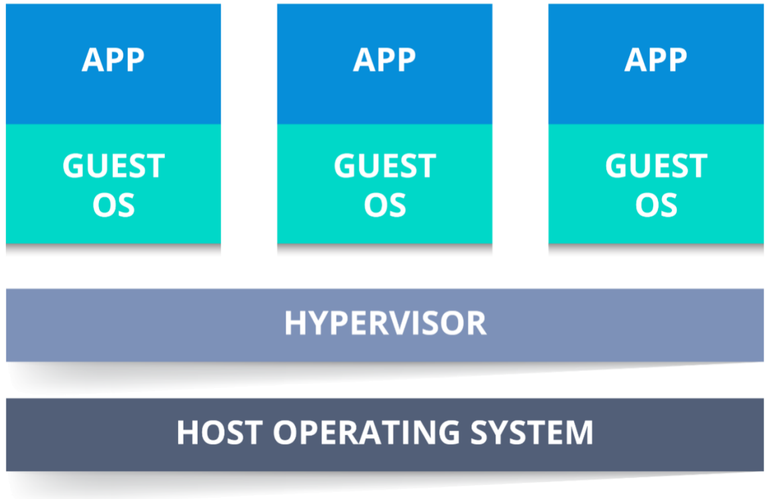
edureka!

Virtualization

What is virtualization ?

- Virtualization is abstraction of underlying physical hardware resources and presenting it in a virtual, logical way to be consumed as a resource
- An underlying physical HW or host is a server, which provides all the required resources
- Virtualization capability is provided by the software called, Hypervisor
- Virtual machine is logical representation of computer system. It is also called “Guest”

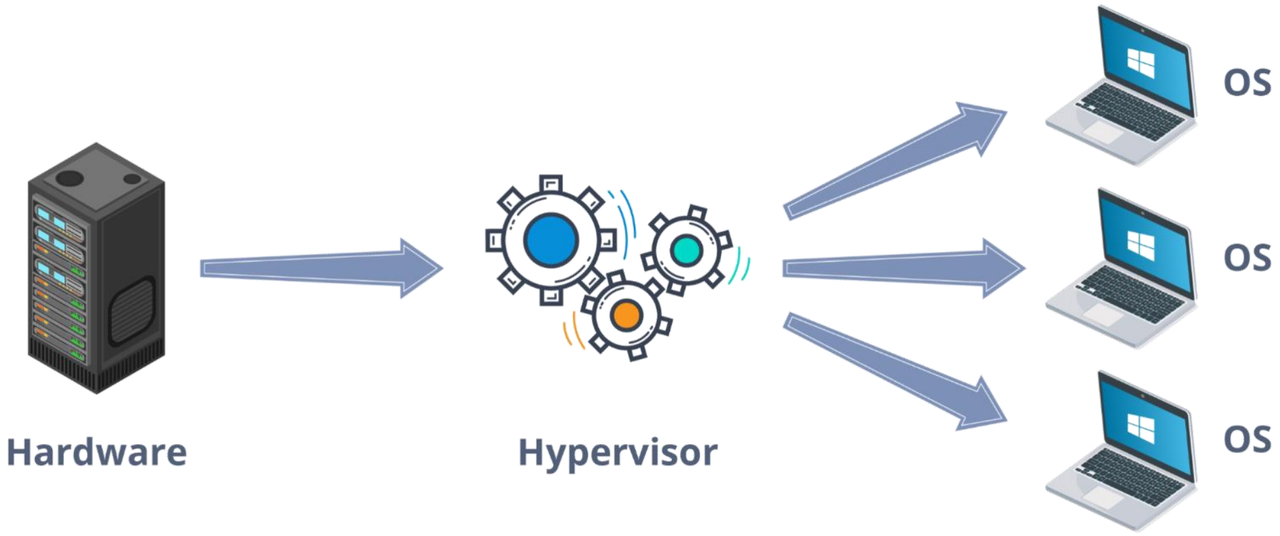
Virtualization



Virtualization - Hypervisor

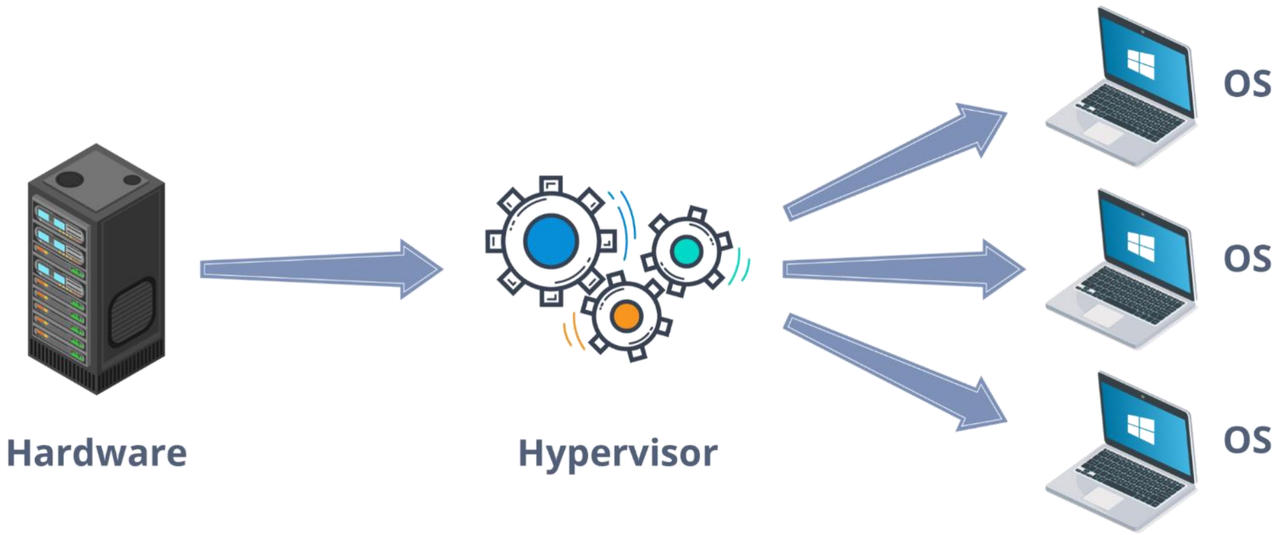
Hypervisor

- Also known as “Virtual Machine Monitor”
- Is the software that runs and create virtual machine.
- Hypervisor runs on a physical server and it is called “host”



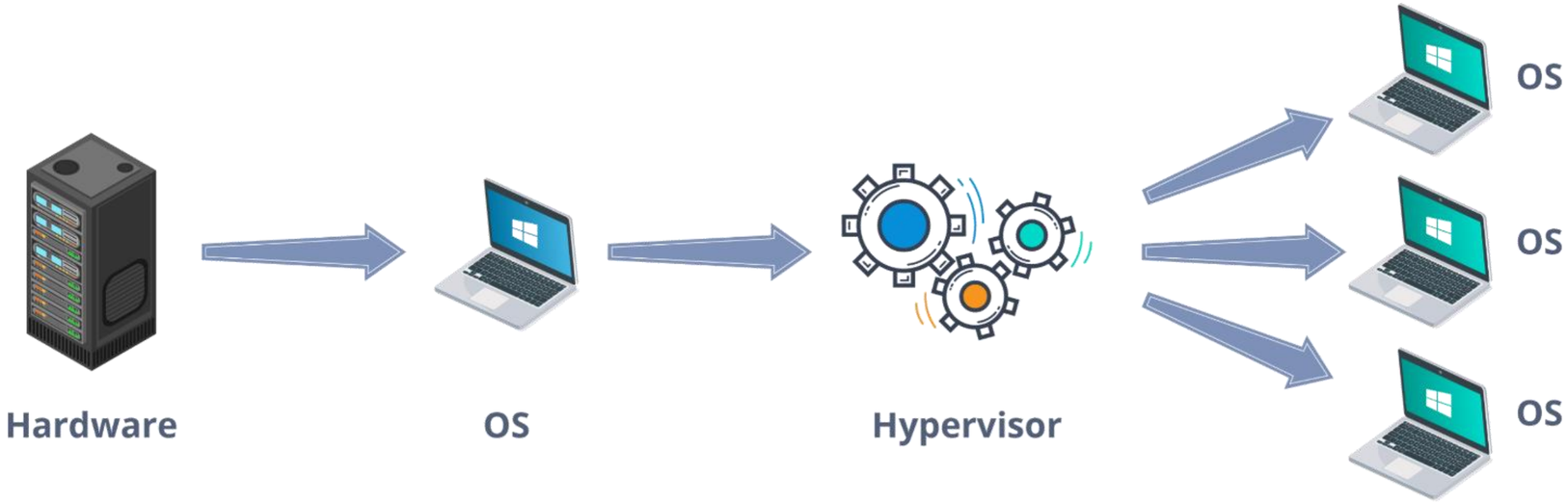
Virtualization – Types of Hypervisor

Type 1: Run directly on the bare-metal or system hardware.
Eg: VMware ESXi



Virtualization – Types of Hypervisor

Type 2: Runs on host operating system.
Microsoft hyper-V, Linux KVM

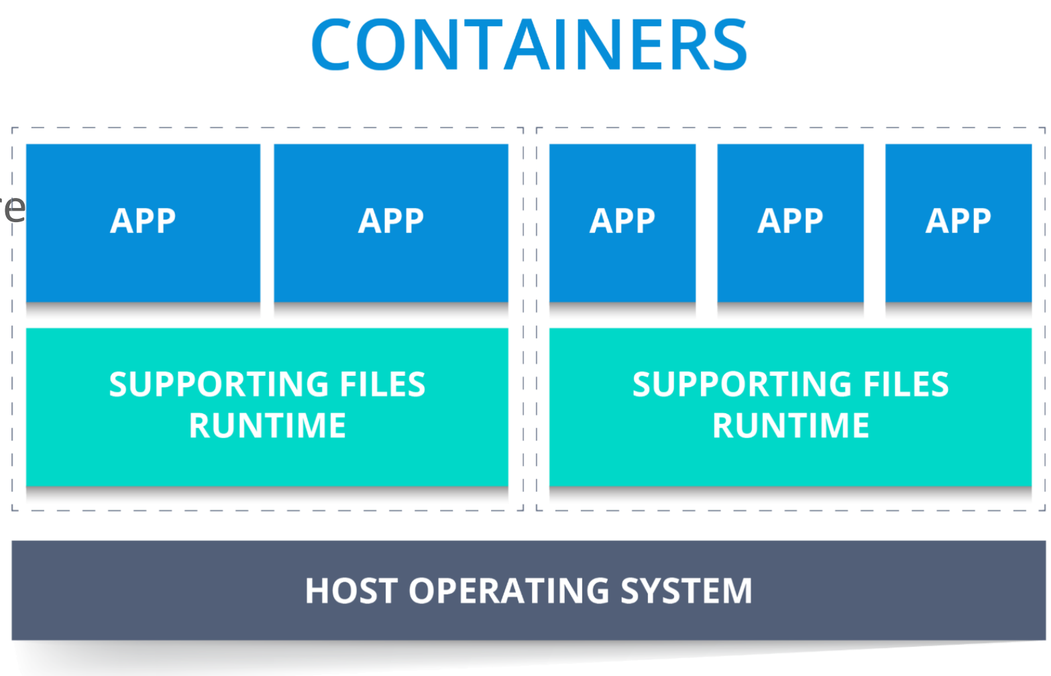


Containerization

edureka!

What is Containerization?

- Containerization is a software which enables operating system to create multiple isolated user-space for applications
- Operating System could be running on physical hardware or Virtual Machine



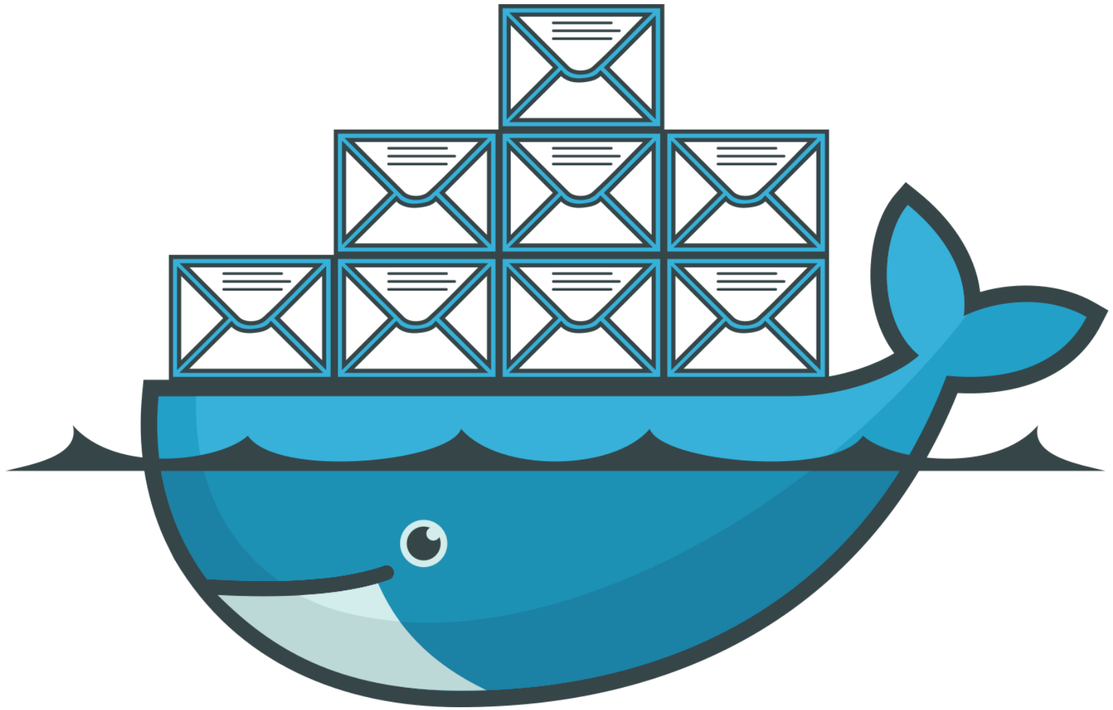
Virtualization V/S Containerization

	Containerization	Virtualization
Flexibility 01	It is easy to move software from one computing environment to another. Eg. from Dev to Staging to Production.	Virtual machine guest is more tightly tied to the underlying hypervisor and movement of application from Dev to Staging to Production environment requires more effort.
Efficient 02	Containers consumes less CPU, memory and disks.	Applications running on Virtualization requires full stack of resource and result in more waste of resources
Management 03	Containers are more in-line with the devops approach. It is more agile and easy to manage than virtualization.	Virtualization management is also easy but when it comes to port applications between different environment then it requires more efforts.

edureka!

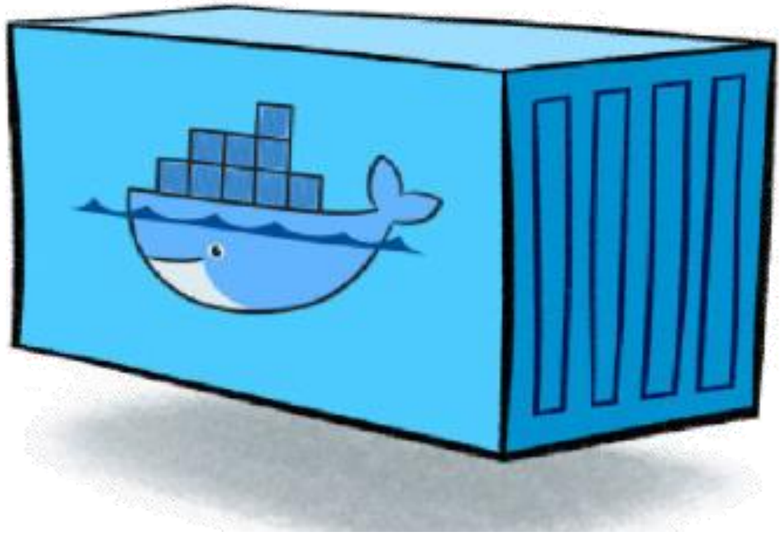
Introduction to Containers

Containers are type of virtualization technology which uses host operating system kernel to run multiple guest instances.



Introduction to Containers

- Each guest instance is called a container
- Each container has its own
  - Root filesystem
  - Network ports
  - Processes
  - Devices
  - Memory



Container we create interacts with the operating system kernel (your operating system could be your physical server, a VM or cloud). Container users certain features of the kernel to create an isolated environment/platform. Within each container we install application and all the libraries that application depends on)

Docker - Essentials and aShort Recap

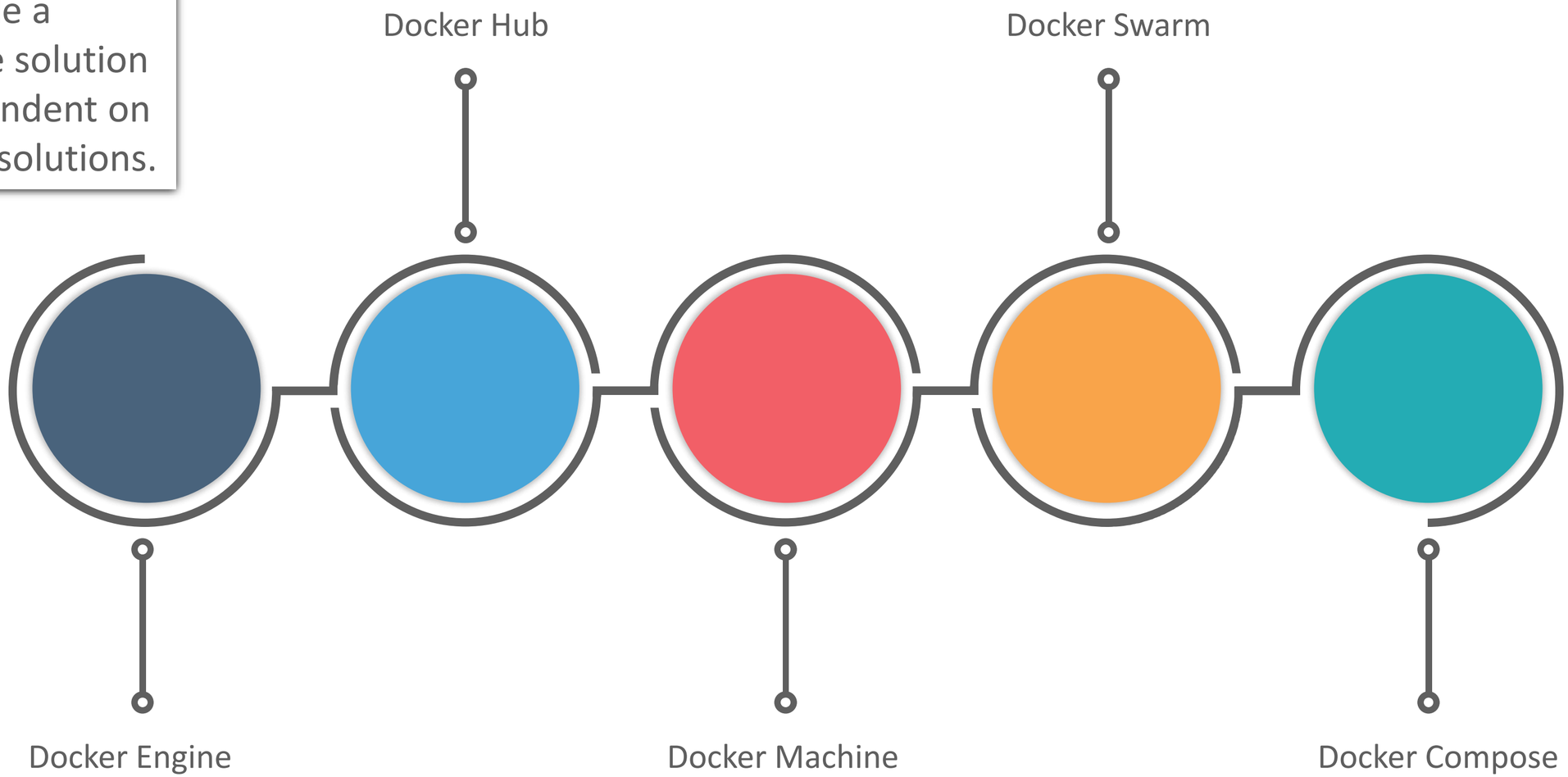
edureka!

Docker - Essentials and a Short Recap

1. It is a software which provides operating system level virtualization, called Containers.
2. It is developed and primarily managed by Docker, Inc.
3. It uses Linux kernel features like cgroups, namespaces , chroot and others to provide resource isolations.

Docker - Essentials and a Short Recap

To provide a complete solution it is dependent on multiple solutions.



Docker Components

Docker Engine

Docker Hub

Docker Machine

Docker Swarm

Docker Compose

Also called Docker daemon or runtime

This is the program we install on each Docker host to provide all the docker services. (consider as shipping yard).

Till the time Docker daemon/runtime (on which Docker is running) is same, application will run irrespective of the environment it is running on (laptop , cloud or on a server)

edureka!

Other Docker Components

- Docker Engine
- **Docker Hub**
- Docker Machine
- Docker Swarm
- Docker Compose

- We pull images from repository
- This “repository” resides in registry.
- The public registry of Docker is on Docker hub (hub.docker.com)

Other Docker Components

- Docker Engine
- Docker Hub
- **Docker Machine**
- Docker Swarm
- Docker Compose

Docker Machine is a tool that automatically provisions Docker hosts and install the Docker Engine on them

It does following operations:

- Create additional hosts on your own computer
- Create hosts on cloud providers (eg. AWS)
- Machine creates the server, installs Docker and configures the Docker client.

Other Docker Components

- Docker Engine
- Docker Hub
- Docker Machine
- **Docker Swarm**
- Docker Compose

It is a tool that clusters Docker hosts and schedules containers

It turns a pool of host machines into a single virtual host.

Ships with simple scheduling backend

Supports many discovery backends:

- Hosted discovery
- Etcd
- Consul
- Zookeeper
- Static files

edureka!

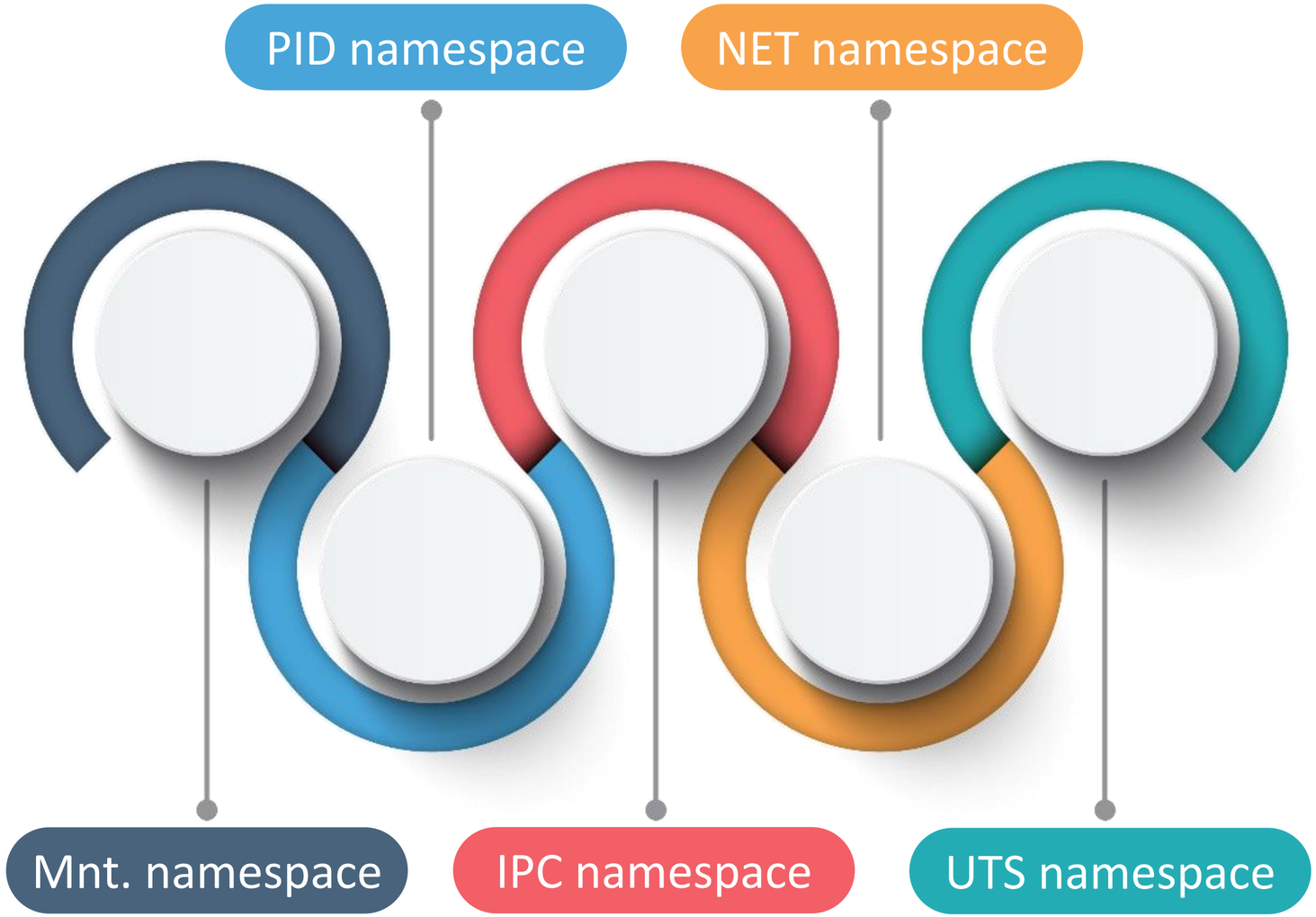
Other Docker Components

- Docker Engine
- Docker Hub
- Docker Machine
- Docker Swarm
- **Docker Compose**

- It is a tool for creating and managing multi container applications
- Containers are all defined in a single file called docker-compose.yml
- Each container runs a particular component/service of your application.
- Container links are defined
- Compose will spin up all your containers in a single command

edureka!

Common Namespace



Common Namespace

1. **PID Namespace**
2. NET Namespace
3. Mnt. Namespace
4. IPC Namespace
5. UTS Namespace

- Isolates the process
- The container is only aware of its native processes
- Its completely aloof to the processes running in different parts of the system
- Host operating system is aware of processes running inside of the container and it assigns them different PID numbers

edureka!

Common Namespace

1. PID Namespace
2. NET Namespace
3. Mnt. Namespace
4. IPC Namespace
5. UTS Namespace

- Network namespace for managing the network stack
- You can add virtual or real devices to the container by assigning assign them their own IP addresses(ip addr)

edureka!

Common Namespace

1. PID Namespace
2. NET Namespace
3. Mnt. Namespace
4. IPC Namespace
5. UTS Namespace

- Isolates filesystem mount points. Eg. Each container can have its own /tmp, /var or even have an entirely different userspace

edureka!

Common Namespace

1. PID Namespace
2. NET Namespace
3. Mnt. Namespace
4. IPC Namespace
5. UTS Namespace

- Isolates certain interprocess communication (IPC)

- The two containers can create shared memory segments and semaphores with the same name, but are not able to interact with other containers memory segments or shared memory

edureka!

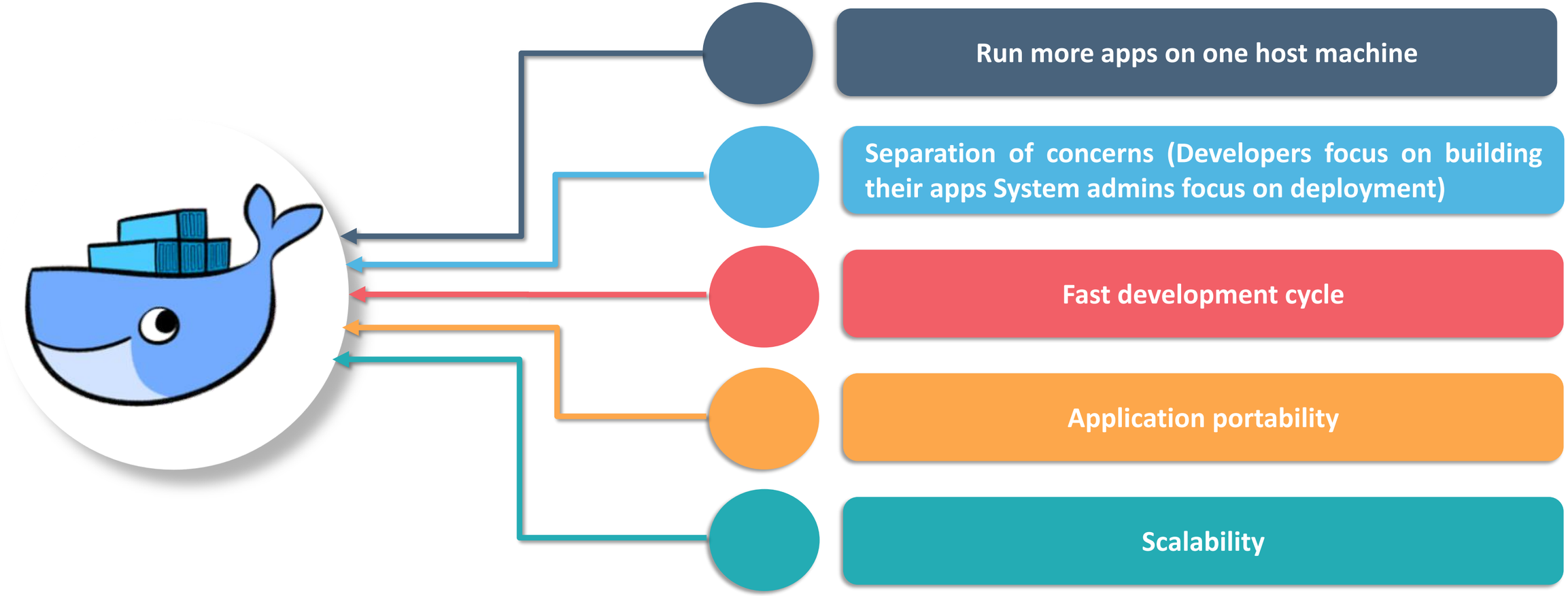
Common Namespace

1. PID Namespace
2. NET Namespace
3. Mnt. Namespace
4. IPC Namespace
5. UTS Namespace

- Isolate two system identifiers – nodename and domainname.
- This isolation allows each container to have its own hostname and NIS domain name
- Isolation is Useful for initialization and configuration scripts based on these names.
- Isolated namespace is called “container”

edureka!

Benefits of Docker



Demo - 1 :Install Docker/Docker-client

edureka!

Demo: Install Docker/Docker-Client

1. sudo apt-get install curl
2. curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
3. sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
4. sudo apt-get update
5. apt-cache policy docker-ce
6. sudo apt-get install -y docker-ce

edureka!

Demo - 2: Launching your first Container

edureka!

Demo: Launching your first Container

- Use the docker run command to pull and execute the docker container

  Command: docker run -it -p 8081:80 -d httpd

- Enter localhost:8081 on your browser to verify

edureka!

Demo - 3: Create and Run a Custom Docker Image

edureka!

Demo: Create and Run a custom Docker Image

- Create a sample html page to view on your httpd home

  Command: nano index.html

- Create a new docker file and add the following

  Command: sudo nano Dockerfile

- Build the dockerfile using docker build command

  Command: sudo docker build . -t demo

- Now run the build docker image

  Command: sudo docker run -it -p 8081:80 -d demo

- Verify by typing localhost:8081 in your browser url

edureka!

Quiz

1. Select which all statement as False for Docker :
   a. It is a cluster solution which provides virtualization ?
   b. It is containerized solution which provides resource isolaton at Operation System level
   c. Container based on docker can run all types of applications

edureka!

Answers

A

1. Select which all statement as False for Docker :
   a. It is a cluster solution which provides virtualization ?
   b. It is containerized solution which provides resource isolaton at Operation System level
   c. Container based on docker can run all types of applications

Answer A, C: Only B is correct.

edureka!

Quiz

2. Fill in the blanks. Each docker container has it’s own :

   a. -----

   b. -----

   c. -----

edureka!

Answers

2. Fill in the blanks. Each docker container has it’s own :

   a. -----

   b. -----

   c. -----

Answer :

1. Root filesystem
2. Network ports
3. Processes

Quiz

3. Fill in the blanks. Provide three commonly used namespace of docker containers and its details:

   a. -----

   b. -----

   c. -----

edureka!

Answers

3. Fill in the blanks. Provide three commonly used namespace of docker containers and its details:

   a. -----

   b. -----

   c. -----

Answer :

1) **PID namespace** – which isolates the process. The container is only aware of its native processes and can not “see” the processes running in different parts of the system. On the other hand, the host operating system is aware of processes running inside of the container, but assigns them different PID numbers
2) **Net namespace** – Network namespace for managing the network stack. You can add virtual or real devices to the container, assign them their own IP addresses. (ip addr)
3) I**PC namespace** – isolates certain interprocess communication (IPC). So, the two containers can create shared memory segments and semaphores with the same name, but are not able to interact with other containers memory segments or shared memory.

YAML

edureka!

What is YAML?

Official Definition : YAML is a human friendly data serialization standard for all programming languages

It is considered as superset of JSON.

Any valid JSON file is also a valid YAML file. But vice-versa is not always true.

What is YAML

"Yet Another Markup Language", or "YAML Ain't Markup Language".

What is YAML?

01 YAML is the main language for creating and managing PODs. in Kubernetes

02 It is used for holding system configuration detail, meta-data, and other settings.

03 Version 1.2 (3rd Edition) is the latest edition released in the year 2009.

04 All the latest and official information are maintained at http://yaml.org/

Basics

- Human Readable
- Uses indentation to define the scope of each block
- Each block entry begins with a dash and a space "- "
- Three "---" dashes or hyphens. This is optional and is used to separate documents within a stream

```
--- !clarkevans.com/^invoice
invoice: 34843
date   : 2001-01-23
bill-to: &id001
    given  : Chris
    family : Dumars
    address:
        lines: |
            458 Walkman Dr.
            Suite #292
        city    : Royal Oak
        state   : MI
        postal  : 48046
ship-to: *id001
product:
    - sku         : BL394D
      quantity    : 4
      description : Basketball
      price       : 450.00
    - sku         : BL4438H
      quantity    : 1
      description : Super Hoop
      price       : 2392.00
tax  : 251.42
total: 4443.52
comments: >
    Late afternoon is best.
    Backup contact is Nancy
    Billsmer @ 338-4338.
```

Basics

Three "..." dots.

End-of the Document

"#" hash or pound sign.

Comments

Does not accept "Tab", and uses white spaces

Structures

Data Structure of YAML can be represented as :

Scalars

used for strings / numbers

Mapping

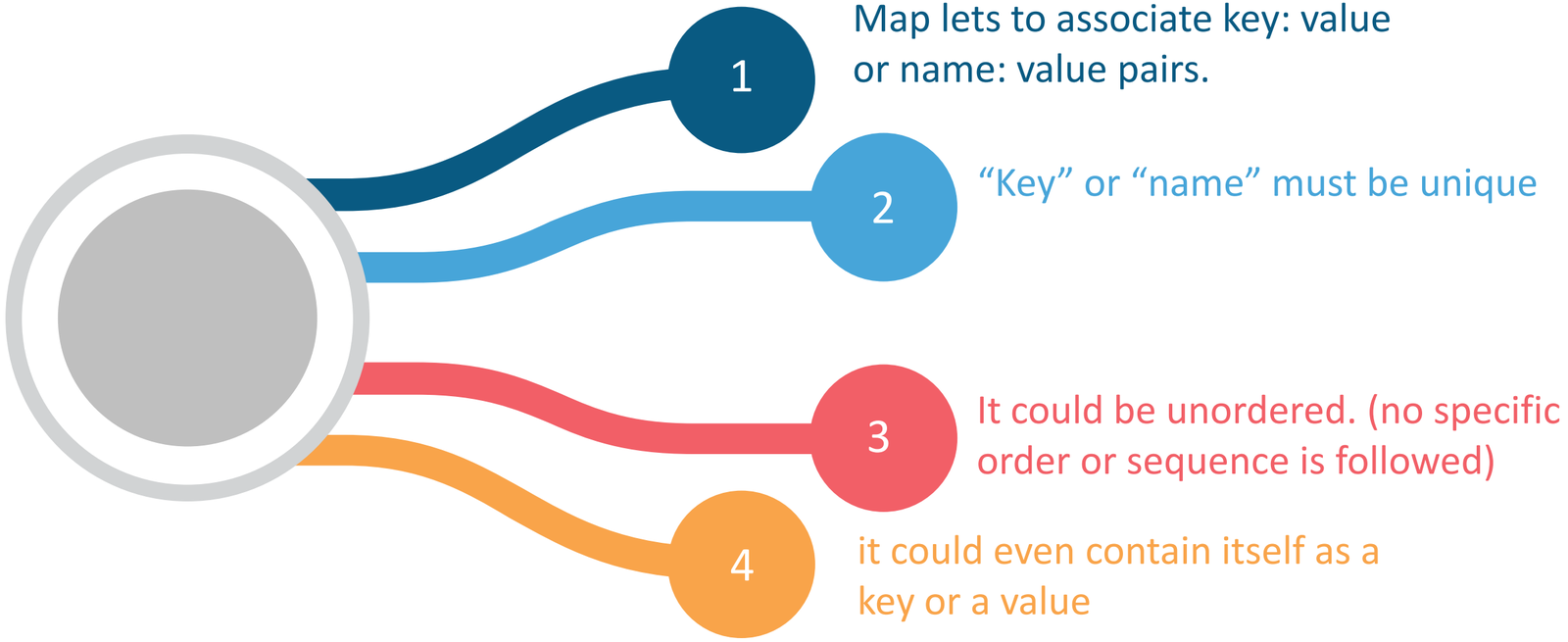
also known as hashes / dictionaries

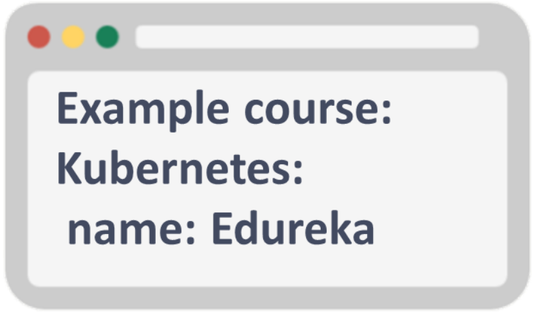
Sequences

also known as arrays / lists

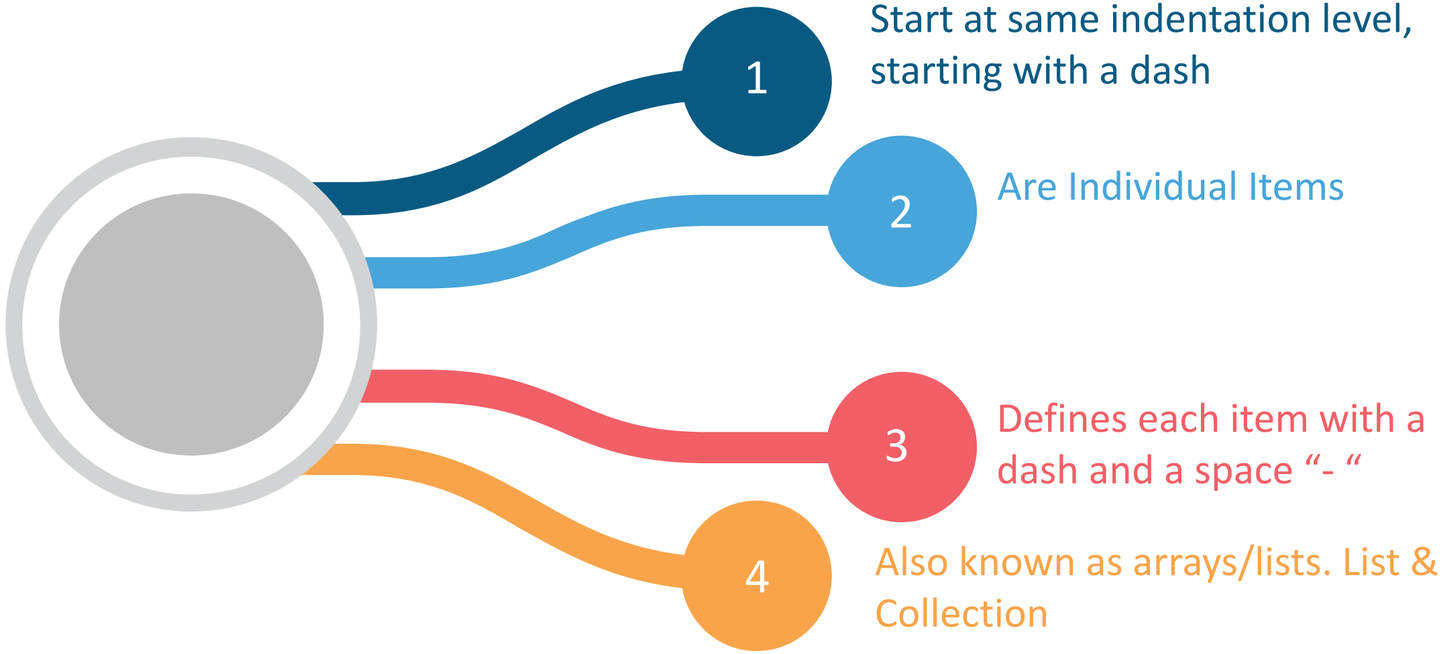
Scalars

Mapping





Sequences



Example
- container
- docker
- kubernetes

Syntax

In real life, you can mix-match all the data structure to achieve the desired result.

- maps, which are groups of key-value or name-value pairs
- lists, which are individual items
- maps of maps
- maps of lists
- lists of lists
- lists of maps

Let’s understand each of them with syntax.

Syntax

maps, which are groups of key-value or name-value pairs

```
---                         # Line separater
apiVersion: v2              # This is Map. Notice the space after ": "
kind: container             # ":" is invalid syntax
```

- Number of space do not matter. Minimum should be 1 and should be structured properly.
  CAUTION : NEVER use tabs in a YAML file.

lists are individual items. A list can have any number of items

```
- version                  # After dash there is minimum one whitespace "- "
- 1.2
```

edureka!

Syntax

maps of maps

```
---                         # Line separater
apiVersion: v2              # This is Map. Notice the space after ": "
kind: container             # ":" is invalid syntax
metadata:                   # key metadata is initiated. ":" is right after the key
  name: web-service         # There is whitespace between ": value"
```

- Here key "metadata" has value as "name".
- "Name" is nested(maps of maps) and have value as "web-service"

Syntax

maps of lists

```
spec:
  containers:          # List of container object which is name, image and list of ports
  - name: web-proxy    # This shows how you can put map of lists
    image: nginx
    ports:
      - containerPort: 80
```

- Also if we notice, “ports” is nested map that list containerPort and its value.

Syntax

```
--- !clarkevans.com/^invoice
invoice: 34843
date   : 2001-01-23
bill-to: &id001
    given  : Chris
    family : Dumars
    address:
        lines: |
            458 Walkman Dr.
            Suite #292
        city    : Royal Oak
        state   : MI
        postal  : 48046
ship-to: *id001
product:
    - sku         : BL394D
      quantity    : 4
      description : Basketball
      price       : 450.00
    - sku         : BL4438H
      quantity    : 1
      description : Super Hoop
      price       : 2392.00
tax  : 251.42
total: 4443.52
comments: >
    Late afternoon is best.
    Backup contact is Nancy
    Billsmer @ 338-4338.
```

Scalar

List & Maps

Combining everything we did so far.

YAML Validators

YAML validator are used to check if the YAML file is correct or not

- Few of the widely used validator are:
  - YAML Validator
  - CodeBeautify YAML Validator
- Text editors that you can use

Sublime Text

Atom

Demo - 4: Write a Simple Docker-Compose File Using YAML

edureka!

Quiz

1. Which is the latest release of YAML
   a. 7.8
   b. 2017.3
   c. 1.2
   d. Version 2015.3.1

Answers

1. Which is the latest release of YAML
   a. 7.8
   b. 2017.3
   c. 1.2
   d. Version 2015.3.1

Answer C:

Quiz

2. Define any two structure of YAML in one line
   a. Scalars
   b. Mapping
   c. Sequences

edureka!

Answers

A

2. Define any two structure of YAML in one line
   a. Scalars
   b. Mapping
   c. Sequences

Answer:

a) Scalars : used for strings / numbers
b) Mapping : also known as hashes / dictionaries. Used to hold key-value pair
c) Sequences also known as arrays / lists. Used to hold components

edureka!

Quiz

3. Which of the following is False?
   a. YAML stands for : “Yet Another Markup Language”, or “YAML Ain’t Markup Language”.
   b. YAML is a human friendly Object Oriented programming languages.
   c. JSON is superset of YAML
   d. YAML accepts “Tab” to accept space for indentation.

Answers

A

3. Which of the following is False?

   a. YAML stands for : “Yet Another Markup Language”, or “YAML Ain’t Markup Language”.

   b. YAML is a human friendly Object Oriented programming languages.

   c. JSON is superset of YAML

   d. YAML accepts “Tab” to accept space for indentation

Answer B,C,D:

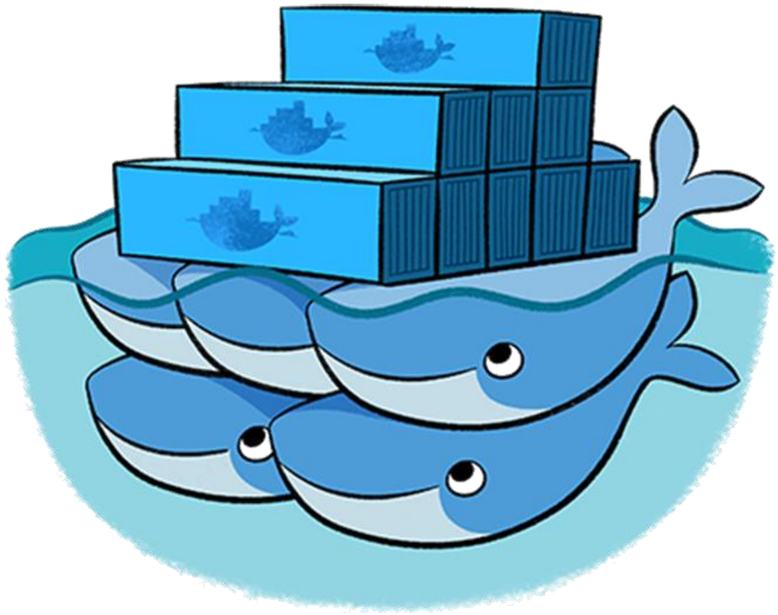
edureka!

Container Orchestration

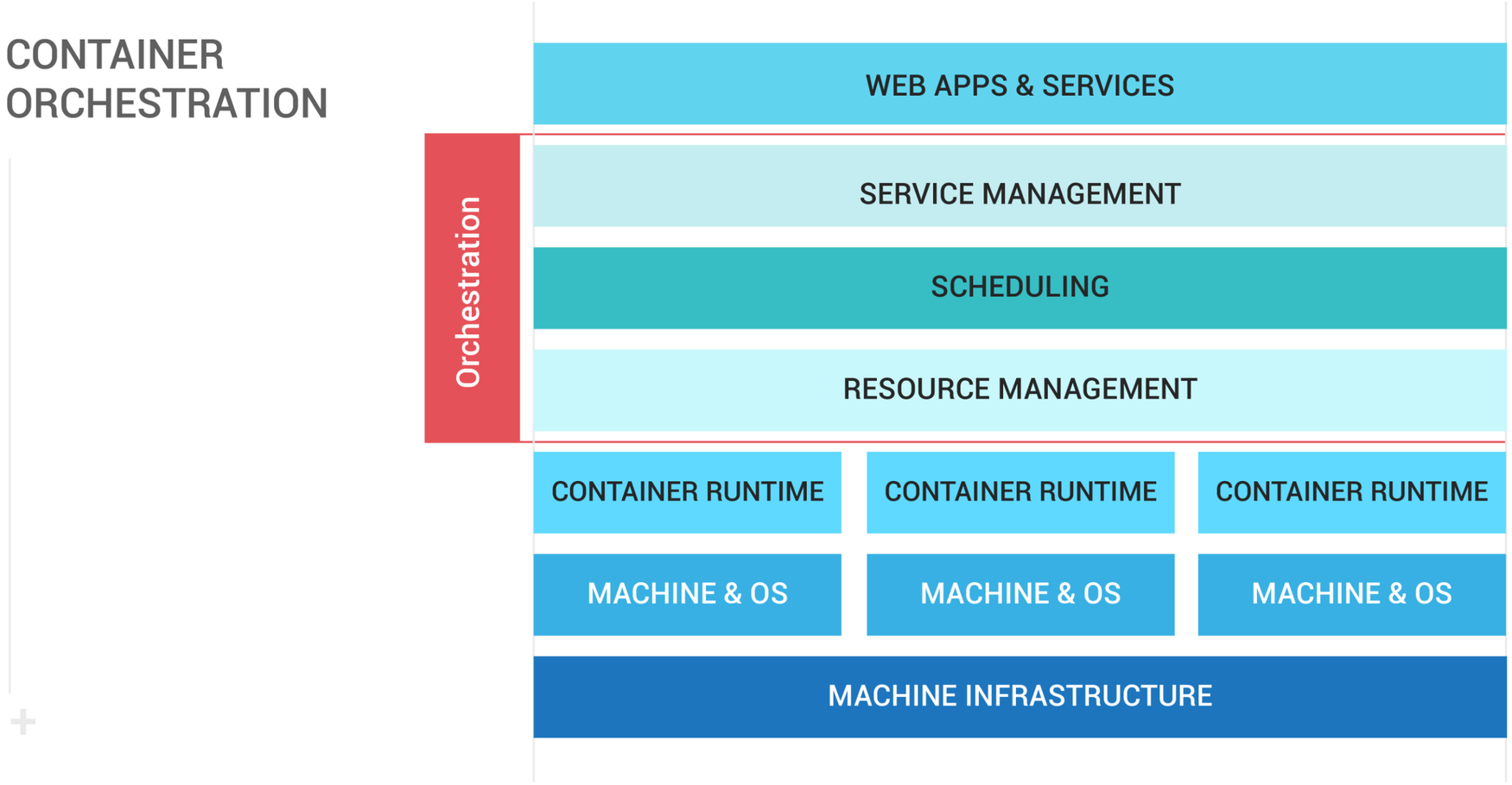
edureka!

Why Container Orchestration?

- Container orchestration manages availability, scaling and networking of the containers
- It helps in monitoring the cluster i.e., group of hosts
- It helps in managing the timing of container creations
- It helps in container configuration in order to allow containers to communicate with one another

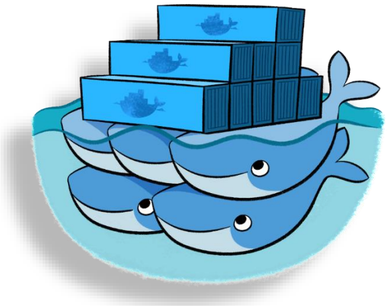


Container Orchestration



The Top 3 Container Orchestrators

The Top 3 Container Orchestrators are :

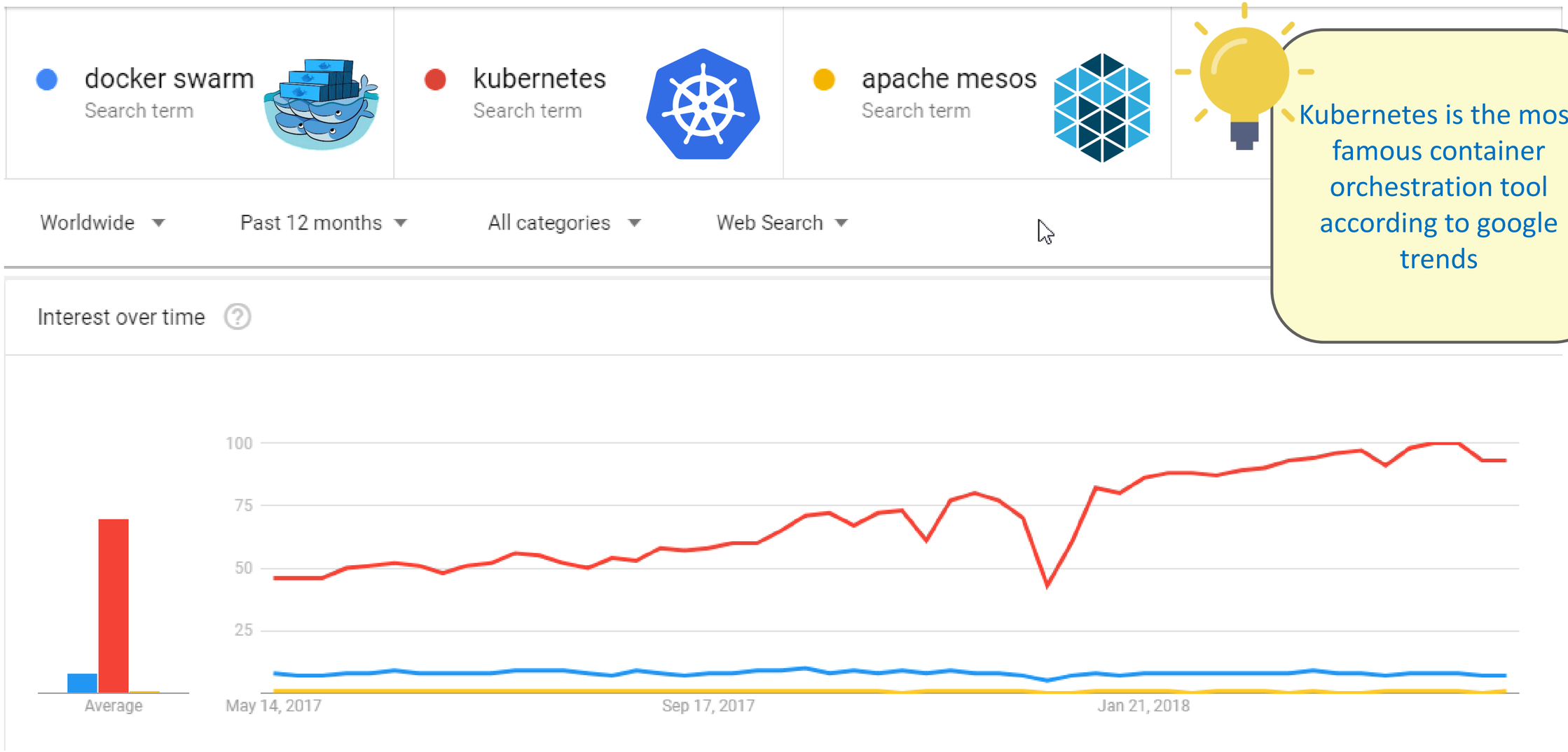


Docker Swarm

Mesos

Kubernetes

The Top 3 Container Orchestrators

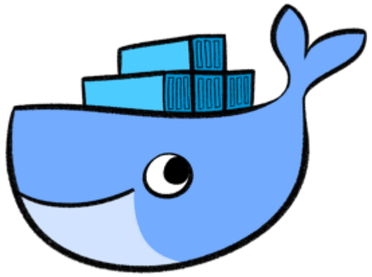


Kubernetes is the most famous container orchestration tool according to google trends

Docker Swarm Vs Kubernetes

Docker Swarm

- Services are discoverable easily through the whole network in Docker Swarm
- It can easily run with other docker tools
- Local volume can be shared easily
- It provides quick container deployment as well as scaling even in very large clusters



Kubernetes

- Containers can be defined as services which makes them easily discoverable in Kubernetes
- It can easily run on any Operating System
- Volume is shared within the pods
- It provides strong guarantees at the expense of speed to cluster states

What is Kubernetes?

edureka!

What is Kubernetes?

- **Kubernetes is an open-source, portable platform for automating deployment, scaling and management of containerized workloads and applications**
- It groups containers that make up an application into logical units for easy management and discovery
- Hence, it's called a container orchestration tool

Kubernetes - History

Refer: https://github.com/kubernetes/kubernetes

- Baked and cooked at Google
- It was donated to CNCF (Cloud Native Computing Foundation) in 2014. And since then it is managed by CNCF
- Written in Go / Golang

What Kubernetes is NOT

edureka!

What Kubernetes is not

1. Kubernetes is not a container. It is an orchestration software which manages containers.

2. It is not the solution to all problems. Containers cannot replace the physical standalone deployment of Servers or Virtual machines.

3. It does not limit the type of applications supported. Kubernetes aims to support an extremely diverse variety of workloads.

Quiz

1. Define : Type 1 and Type 2 hypervisor. Give examples of each.

edureka!

Answers

1. Define : Type 1 and Type 2 hypervisor. Give examples of each.

Answer:

Type 1 : Run directly on the bare-metal or system hardware. VMware ESXi
Type 2: Runs on host operating system. Eg: Microsoft hyper-V

Quiz

2. Which of the following statement is True?
   a. Kubernetes is a virtualization solution
   b. Kubernetes is an orchestration software for containers
   c. Kubernetes also provides containerized solution
   d. Kubernetes & k8s are two different softwares doing the similar work.

edureka!

Answers

A

2. Which of the following statement is True?

   a. Kubernetes is a virtualization solution
   b. Kubernetes is an orchestration software for containers
   c. Kubernetes also provides containerized solution
   d. Kubernetes & k8s are two different softwares doing the similar work.

Answer B:

edureka!

Summary

- In this module, you should have learnt:
  - Recap of Docker
  - YAML and its syntax
  - Virtualization?
  - Containerization?
  - Basics of Kubernetes

Questions

edureka!

Thank You

For more information please visit our website
www.edureka.co