Docker (Noted) - For Beginners

1. Introduction

Docker is a platform designed to develop, ship, and run applications in isolated environments called containers. It simplifies application deployment by packaging code and dependencies together, ensuring consistent performance across different environments.

Why Use Docker?

- **Portability:** Run applications across different environments without modifications.
- **Efficiency:** Uses less memory and CPU compared to virtual machines.
- **Scalability:** Easily scale up or down using orchestration tools like Kubernetes.

Real-World Example:

Suppose you develop a web application on your local machine. Without Docker, moving it to a production server may break it due to environment differences. With Docker, you package everything (code, libraries, configurations) into a container, ensuring it works the same everywhere.

---

2. Core Concepts

Container

A container is a lightweight, portable, and executable software package that includes everything needed to run an application, such as code, libraries, runtime, and configuration files.

Example:

```
docker run -d -p 80:80 nginx
```

This command runs an NGINX web server in a container and maps port 80 of the container to port 80 of the host machine.

Image

An image is a snapshot of a container, containing the application code, dependencies, and configurations.

Example:

docker pull ubuntu

This command pulls an Ubuntu image from Docker Hub.

Dockerfile

A Dockerfile is a text file containing instructions to build a Docker image.

Example:

FROM ubuntu
RUN apt-get update
CMD ["echo", "Hello, Docker!"]

Container Registry

A container registry is a repository to store and distribute Docker images, such as Docker Hub or AWS ECR.

Example:

docker tag myapp:latest myrepo/myapp:latest

Volume

A volume is a directory or file stored outside the container’s filesystem to persist data.

Example:

docker run -v /host/data:/container/data myapp

Network

A Docker network enables communication between containers or between containers and external services.

Example:

```
docker network create mynetwork
```

---

3. Workload Management

Docker Compose

A tool to define and run multi-container applications using a YAML file.

Example:

```
version: '3'
services:
  web:
    image: nginx
    ports:
      - "80:80"
```

Docker Swarm

A native clustering and orchestration tool for Docker to deploy and manage containerized applications across multiple hosts.

Example:

```
docker swarm init
```

Kubernetes Integration

Docker containers can be managed using Kubernetes for large-scale orchestration.

Example:

```
kubectl create deployment myapp --image=myapp
```

---

4. Storage and Configuration

Bind Mounts

Allows mounting host machine directories into containers.

Example:

docker run -v /home/user/data:/data myapp

Volumes

Persist container data even after container restarts.

Example:

docker volume create myvolume

Secrets Management

Manage sensitive data like API keys securely.

Example:

docker secret create mysecret myfile.txt

---

5. Networking

Bridge Network

Default network for containers to communicate with each other on the same host.

Example:

docker network inspect bridge

Host Network

Containers use the host machine's network for high performance.

Example:

docker run --network host myapp

Overlay Network

Used in Docker Swarm to enable communication across hosts.

Example:

docker network create -d overlay myoverlay

Port Mapping

Expose container ports to the host machine.

Example:

docker run -p 8080:80 myapp

---

6. Security

Docker Daemon

Manages Docker containers, images, networks, and volumes.

Namespace Isolation

Provides isolation of processes between containers.

Control Groups (cgroups)

Limits the resource usage (CPU, memory) of containers.

Docker Content Trust (DCT)

Ensures only verified images are pulled and used.

Example:

docker trust inspect myapp

---

7. Scaling and Management

Container Orchestration

Scale and manage containers using Docker Swarm or Kubernetes.

Load Balancing

Distribute incoming traffic across multiple containers.

Service Discovery

Enable containers to discover and communicate with each other.

Container Logs

Capture and monitor logs.

Example:

docker logs myapp

---

8. Additional Features

Docker Build

Build Docker images from a Dockerfile.

Example:

docker build -t myapp .

Docker Push/Pull

Push or pull images from a registry.

Example:

docker push myrepo/myapp

Docker Run

Create and start a container from an image.

Example:

docker run -d myapp

Docker Stop/Remove

Stop and remove running containers.

Example:

docker stop myapp

Docker Inspect

Retrieve detailed information about containers, images, volumes, and networks.

Example:

docker inspect myapp

Conclusion

Docker significantly simplifies application deployment by packaging code, configurations, and dependencies into containers. With proper understanding and practice, mastering Docker will elevate your DevOps skills and improve your deployment efficiency.

Also, Check below quick content to upskill!!

AWS Hands-on Labs (Guide)
https://techyoutube.com/index.php/2023/12/26/24-aws-hands-on-labs-elevate-your-expertise-now/

Azure Hands-on Labs (Guide)
https://techyoutube.com/index.php/2024/01/26/12-azure-hands-on-labs-elevate-your-expertise-now/

DevOps & Cloud Projects Ideas
https://techyoutube.com/index.php/category/devops-cloud-projects/

DevOps FREE Quizzes (Test & Learn)
https://techyoutube.com/index.php/category/quiz/devops-quiz/

Kubernetes - Interview (Questions & Answers)
https://techyoutube.com/?s=kubernetes+interview

SRE (Site Reliability Engineer) - Questions & Answer
https://techyoutube.com/index.php/category/devops/sre-interview-q-a/

🔥 Free Courses : Kubernetes (Enroll Now) 🔥

https://techyoutube.com/index.php/2024/03/02/free-courses-kubernetes-enroll-now/

Terraform Content
https://techyoutube.com/index.php/category/devops/terraform/

Hope you find this document helpful for your Azure Learning.

For more such content you can check : https://techyoutube.com/

Now, to Support, just follow me on below socials (No Cheating Please)

Telegram: https://t.me/LearnDevOpsForFree

Twitter: https://twitter.com/techyoutbe

Youtube: https://www.youtube.com/@T3Ptech