docker

Docker Roadmap 2025

LAHIRU
LIYANAPATHIRANA

What Is Docker?

Docker is an open-source containerization platform that allows developers to package applications and their dependencies into lightweight, portable, self-sufficient units called containers.

These containers run consistently across different environments—whether on a local machine, a server, or the cloud.

Unlike virtual machines, Docker containers share the host OS kernel, making them faster, more efficient, and scalable.

Why Use Docker?

- **Portability:** Containers work on any OS with Docker installed.
- **Consistency:** Eliminate environmental discrepancies.
- **Isolation:** Apps run in sandboxed environments.
- **Scalability:** Easily replicate containers for load balancing.
- **Fast Deployment:** Spin up containers in seconds and integrate with CI/CD pipelines for automated workflows.

docker

Pre-Requisites

- CLI Basics (Terminal/Shell)
- Linux Fundamentals (Filesystem, Processes)
- Git & Version Control
- Basic Networking (Ports, IPs, Protocols)

docker

Beginner

LAHIRU
LIYANAPATHIRANA

docker

Core Concepts

- Virtualization Concepts (VMs, Hypervisors)
- Containerization Fundamentals
- Virtual Machines vs Containers

Docker Architecture

- Docker Architecture
- Docker Client
- Docker Daemon
- Docker Registry and Docker Hub
- Image Repositories
- Docker Desktop

docker

Core Docker Components

- Container
- Image
- Networks
- Volumes

Docker Installation

- Docker Installation (Windows, macOS, Linux)
- Docker Desktop (Windows, macOS)
- Docker Engine (Linux)

Basic Docker Operations

- Docker CLI Essentials
- Docker Image Management
  - Image Pulling (docker pull)
  - Image Listing (docker images)
  - Image Deletion (docker rmi)
- Docker Container Management
  - Container Listing (docker ps)
  - Container Stopping (docker stop)
  - Container Starting and Restarting (docker start/restart)
  - Container Removal (docker rm)
  - Container Logs (docker logs)

Basic Docker Operations

- Dockerfile Basics
  - Dockerfile Definition
  - Basic Instructions: FROM, RUN, CMD, ENTRYPOINT, COPY, EXPOSE, ENV, WORKDIR
  - Simple Image Building (docker build)
- Docker Run
- Building and Running The First Container

docker

Storage & Volumes

- Bind Mounts (v flag)
- Named Volumes

Networking

- Default Bridge Network
- Port Mapping (p 8080:80)

Docker Compose

- Basic docker-compose.yml Structure (Services, Networks, Volumes)
- Commands: docker-compose up, down, logs

docker

Security

- Running Containers as Non-Root Users
- Avoiding Privileged Containers

docker

Intermediate

LAHIRU
LIYANAPATHIRANA

docker

Concepts

- Namespaces (PID, NET, MNT) & Control Groups (cgroups)
- Union File System (Image Layering)
- OCI Standards (Image/Container Specs)

Intermediate Docker Operations

- Image Layers & Build Context
- Multi-Stage Builds (Reduce image size)
- Dockerfile:
  - ARG, VOLUME, USER, HEALTHCHECK, ENTRYPOINT

Storage & Volumes

- tmpfs Mounts (In-memory storage)
- Storage Drivers (overlay2, aufs)

Networking

- Custom Bridge Networks
- Host/None Networks
- DNS Configuration

Docker Compose

- Environment Variables (.env files)
- Service Dependencies (depends_on)
- Resource Limits (CPU/Memory)

Security

- Capabilities Management (–cap-add/drop)
- Seccomp/AppArmor Profiles (Restrict syscalls)
- Image Scanning Tools (Trivy, Snyk)

Orchestration (Docker Swarm)

- Swarm Mode Basics (docker swarm init)
- Service Scaling (docker service scale)
- Stack Deployments (docker stack deploy)
- Overlay Networks

docker

Monitoring & Logging

- Logging Drivers (JSON, Syslog)
- Health Checks

Production & Enterprise

- CI/CD Pipelines (Jenkins, GitLab CI)

docker

Advanced

LAHIRU
LIYANAPATHIRANA

Concepts

- containerd (Docker's default container runtime)
- Security Namespaces (User, Mount, UTS)

Advanced Docker Operations

- Advanced Image Cache Management
- Buildx for Multi-Architecture Images
- Advanced BuildKit Features
- Image Signing (Docker Content Trust)

docker

Storage & Volumes

- Volume Plugins (e.g., AWS EBS)
- Storage Architecture Deep Dive (Copy-on-Write, Snapshotting)

Networking

- Macvlan/Overlay Networks
- Network Plugins (Calico, Weave)
- Network Troubleshooting

Docker Compose

- Compose Override Files
- Secrets/Configs Management

docker

Security

- SELinux/AppArmor Integration
- Runtime Protection (Security Contexts)
- Secrets Encryption

Orchestration (Docker Swarm)

- Raft Consensus
- Rolling Updates/Rollbacks
- Placement Constraints

Kubernetes Integration

- Deploying Docker Containers in Kubernetes Pods
- Docker Desktop's Built-in Kubernetes Cluster

Monitoring & Logging

- Prometheus + Grafana Integration
- Centralized Logging (ELK Stack)

Production & Enterprise

- Blue-Green/Canary Deployments
- Docker Enterprise Tools (UCP, DTR)

Other Advanced Concepts

- Docker API/SDKs (Python, Go)
- Custom Plugins (Network, Storage)
- Rootless Docker
- Cross-Platform Builds

docker

How Can You Improve?

- Learn the fundamentals
- Build simple projects
- Move on to complex projects
- Learn best practices
- Practice, practice, practice

Did You Find This Post Useful?

Stay Tuned for More Docker Related Posts Like This

LAHIRU LIYANAPATHIRANA