docker

Module Zero

Course Overview

Docker’s vision

Increase the time developers spend on innovation, and decrease the time they spend on everything else

Schedule

Day One

- 8:00 AM - 8:15 AM: Course Overview
  (15 minutes)
- 8:15 AM - 9:00 AM: Module 1: Introduction to Docker and Containers
  (45 minutes)
- 9:00 AM - 9:45 AM: Module 2: Introduction to Docker Desktop
  (45 minutes)
- 9:45 AM - 10:00 AM: Morning Break
  (15 minutes)
- 10:00 AM - 11:00 AM: Module 3: Docker Desktop - Developer Focused
  (1 hour)
- 12:15 PM - 1:15 PM: Lunch Break
  (60 minutes)
- 1:15 PM - 2:15 PM: Module 5: Image Deep Dive (Part 1)
  (1 hour)
- 2:15 PM - 2:30 PM: Afternoon Break
  (15 minutes)
- 2:30 PM - 3:30 PM: Module 5: Image Deep Dive (Part 2)
  (1 hour)
- 3:30 PM - 3:45 PM: Break
  (15 minutes)
- 3:45 PM - 4:00 PM: Wrap-up/Q&A
  (15 minutes)

Day Two

- 8:00 AM - 9:15 AM: Module 6: Best Practices
  (1 hour 15 minutes)
- 9:15 AM - 9:30 AM: Morning Break
  (15 minutes)
- 9:30 AM - 10:45 AM: Module 7: Compose and Testcontainers
  (1 hour 15 minutes)
- 10:45 AM - 11:00 AM: Break
  (15 minutes)
- 11:00 AM - 12:00 PM: Module 8: Docker in CI/CD
  (1 hour)
- 12:00 PM - 1:00 PM: Lunch Break
  (60 minutes)
- 1:00 PM - 2:00 PM: Module 9: Docker Networking
  (1 hour)
- 2:00 PM - 2:15 PM: Afternoon Break
  (15 minutes)
- 2:15 PM - 3:45 PM: Module 10: Customer Topics
  (1 hour 15 minutes)
- 3:45 PM - 4:00 PM: Final Q&A/Closing Remarks
  (15 minutes)

docker

Module Detail

Module One

Introduction to Docker and Containers (45 mins)

- Docker's Vision
  - Increase developer productivity by reducing non-innovative tasks.
- Before vs. After Containerization
  - Highlights the improvements containers bring to development workflows.
- Shipping Software
  - Comparisons to real-world shipping containers—standardization boosts efficiency.
- Container Specifications
  - Describes the Open Container Initiative (OCI) standards for containers.
- Container Benefits
  - Improved developer productivity, faster deployments, and enhanced portability across environments.

Module Two

Introduction to Docker Desktop (45 mins)

- Components of Docker Desktop
  - Explanation of Docker CLI, Compose, Kubernetes, and Buildkit.
- Docker Desktop Architecture
  - How Docker Desktop integrates components via a VM for consistency across OS platforms.
- Networking and Volume Management
  - Overview of managing Docker networks and volumes using Docker Desktop.
- Docker Build Cloud and Docker Scout
  - Features for accelerating build processes and analyzing container images.
- Advantages of Docker Desktop
  - Simplified and integrated developer experience with built-in security and governance features.

Module Three

Docker Desktop - Developer Focused (75 mins)

- Inner Loop & Outer Loop
    - Workflow stages from development (inner loop) to production (outer loop).
- Shift Left
    - Moving testing and validation earlier in the development process.
- Dockerfile Enhancements
    - Features like HEREDOC support and using mounts for secret or cache management.
- Docker Init & Compose Watch
    - New tools for rapid container setup and live updates to running containers during development.
- Compose Bridge
    - Integration with Kubernetes for deploying Compose applications in K8s environments.
- Demo
    - Demonstrate inner loop and outer loop workflows with their pros and cons.

Module Four

Docker Security (60 mins)

- Authentication Options
  - Methods for logging into Docker, including SSO, SCIM, and username/password.
- Docker Desktop Security Architecture
  - Key features for securing Docker Desktop environments, including container isolation and hardened VM security.
- Threat Landscape
  - Common threats to Docker environments, including malware and misconfiguration by developers.
- Hardened Docker Desktop
  - Features to enhance container isolation and mitigate threats without disrupting the developer workflow.
- Registry and Image Access Management
  - Controlling which registries and images developers can access to ensure security.

Module Five
Image Deep Dive (90 mins)

- OCI Specifications
  - How Docker images adhere to standards for containerization.
- Image Layering and Union Filesystems
  - Explanation of image layers, how changes are handled, and why whiteout files exist.
- BuildKit and docker buildx
  - Modern build tools for advanced scenarios, including multi-stage and multi-architecture builds.
- Cache Management
  - How Docker manages build caches to improve efficiency in CI/CD pipelines.
- Multi-stage Builds
  - Using multiple stages in Dockerfiles to optimize images and separate build and runtime environments.
- Demo
  - Multi-stage Builds
  - Building images with tags and labels
  - Creating an SBOM

Module Six

Best Practices (75 mins)

- Governance and Containers
  - Importance of maintaining consistency and security in Docker image creation and usage.
- Repositories, Tags, and Labels
  - Best practices for managing Docker repositories, semantic versioning, and adding metadata with labels.
- Security Considerations
  - Leveraging vulnerability scanning and security best practices in image governance.
- Optimizing Developer and Production Images
  - Tips for creating efficient, secure Docker images for different environments.
- HEALTHCHECK Directive
  - Monitoring the health of containerized applications using Dockerfile directives.
- Demo
  - Image cleanup with Docker Scout
  - Image best practices

Module Seven

Compose and Testcontainers (90 mins)

- Compose Basics
  - Overview of Docker Compose and its commands for managing multi-container applications.
- Cross-service Dependencies
  - Managing service dependencies in Compose files with depends_on and health checks.
- Testcontainers
  - Testing tools for Docker, enabling isolated testing environments.
- Networking in Compose
  - How Compose manages networking and DNS for services within a Compose file.
- Demo
  - Python → Compose → Kubernetes
  - Compose Usage
  - Testcontainers

Module Eight
Docker in CI/CD (60 mins)

- CI/CD Overview
  - Defining Continuous Integration (CI) and Continuous Delivery (CD) and their benefits.
- Managing Drift
  - Handling configuration drift across environments in CI/CD pipelines.
- Docker in CI/CD Pipelines
  - Using Docker to ensure environment consistency, reduce integration risks, and enhance collaboration.
- Docker Outside of Docker (DooD) and Docker in Docker (DinD)
  - Pros and con
- Demo
  - DooD, DinD, and Remote Builds

Module Nine

Docker Networking (60 mins)

- Network Types
  - Overview of different Docker network drivers (Bridge, Host, MACVLAN, Overlay, etc.).
- Best Practices
  - Tips for securing and optimizing container networking.
- Container Communication
  - Techniques for internal name resolution and avoiding external port exposure when communicating between containers.
- Security Considerations
  - Network isolation, TLS for remote contexts, and leveraging Docker's Enhanced Container Isolation (ECI).
- Airgapped Containers
  - Fine-tuned control of ingress/egress in secure environments.

Module Ten

Customer Topics (45 mins)

- Customer chosen discussions

Learning Objectives

What you leave with....

- Fundamental Docker Concepts: Understand the basics of Docker, containers, and the Open Container Initiative (OCI) standards.
- How to Use Docker Desktop: Effectively utilize Docker Desktop for local development, including its integration with Kubernetes and Buildkit.
- Building and Managing Docker Images: Create optimized Docker images using multi-stage builds, labels, tags, and best practices for image governance.
- Docker Compose for Multi-Container Applications: Use Docker Compose to define, run, and manage multi-container applications, including networking and service dependencies.
- Container Networking: Understand Docker's networking types and best practices for securing and optimizing container-to-container communication.

What you leave with....

- Integrating Docker into CI/CD Pipelines: Incorporate Docker into CI/CD workflows, managing configuration drift, and ensuring consistency between environments.
- Image Scanning and Vulnerability Management: Use Docker Scout and Dive to identify and resolve image vulnerabilities and optimize image sizes.
- Debugging Docker Builds and Containers: Diagnose and resolve issues with Dockerfile builds and running containers, including using Docker Debug.
- Containerization Best Practices: Follow best practices for tagging, labeling, health checks, and securing Docker environments in production.
- Deploying Applications to Kubernetes: Convert Docker Compose setups to Kubernetes configurations and deploy applications to Kubernetes.

docker