1. Installation von Docker auf verschiedenen Betriebssystemen
2. Erste Schritte mit Docker: Ausführen des Hello-World-Containers
3. Verwendung der Docker-CLI: Grundlegende Befehle und Optionen
4. Erstellung eines Docker-Images aus einer Dockerfile
5. Starten, Stoppen und Entfernen von Docker-Containern
6. Erstellung und Verwaltung von Docker-Netzwerken
7. Erstellung und Verwaltung von Docker-Volumes
8. Verwendung von Umgebungsvariablen in Docker-Containern
9. Erstellung von benutzerdefinierten Docker-Images
10. Verwendung von Build-Argumenten in Docker-Images

Docker-Images:

1. Verwendung von Docker Hub: Herunterladen und Hochladen von Images
2. Erstellung von Multi-Stage-Docker-Images
3. Verwendung von Multi-Stage-Builds zur Optimierung von Docker-Images
4. Erstellung von Alpine-basierten Docker-Images für eine geringe Größe
5. Verwendung von Docker Compose zum Erstellen und Verwalten von Multi-Container-Anwendungen
6. Verwendung von Docker Registry: Erstellung und Verwaltung einer privaten Registry
7. Verwendung von Docker-Images in CI/CD-Pipelines
8. Verwendung von Docker-Build-Kits zur Erstellung von Images in einer Container-Umgebung
9. Verwendung von Docker Layer Caching zur Beschleunigung des Build-Prozesses
10. Verwendung von Docker Buildx zur Unterstützung verschiedener Plattformen und Architekturen

Docker-Container:

1. Verwendung von Umgebungsvariablen in Docker-Containern
2. Exposition von Ports in Docker-Containern
3. Verwendung von Volumes zur Persistenz von Daten in Docker-Containern
4. Verwendung von Bind-Mounts zur Verbindung von Host-Dateien und -Verzeichnissen mit Containern
5. Verwendung von Named Volumes zur gemeinsamen Nutzung von Daten zwischen Containern
6. Verwendung von tmpfs-Volumes zur temporären Speicherung von Daten in Containern
7. Verwendung von Docker-Labels zur Markierung und Identifizierung von Containern
8. Verwendung von Healthchecks zur Überwachung der Container-Gesundheit
9. Verwendung von Restart-Policies zur Steuerung des Neustarts von Containern
10. Verwendung von Docker-Events zur Überwachung von Container-Ereignissen

Docker-Netzwerke:

1. Verwendung von Bridge-Netzwerken zur Kommunikation zwischen Containern auf einem Host
2. Verwendung von Host-Netzwerken zur direkten Verbindung von Containern mit dem Host-Netzwerk
3. Verwendung von Overlay-Netzwerken zur Kommunikation zwischen Containern in verschiedenen Hosts
4. Verwendung von Macvlan-Netzwerken zur Zuweisung von MAC-Adressen an Container
5. Verwendung von IPv6-Netzwerken in Docker
6. Verwendung von Netzwerk-Namespaces zur Isolierung von Containern
7. Verwendung von DNS-Auflösung in Docker-Netzwerken
8. Verwendung von Proxy-Netzwerken zur Verbindung von Containern mit externen Diensten
9. Verwendung von Network-Policies zur Steuerung des Netzwerkverkehrs zwischen Containern
10. Verwendung von External-DNS zur dynamischen DNS-Registrierung von Containern

Docker-Sicherheit:

1. Sicherer Umgang mit Docker-Containern: Best Practices und bewährte Methoden
2. Absicherung von Docker-Images: Verwendung von Content Trust und digitalen Signaturen
3. Isolierung von Containern mit Namespaces und cgroups
4. Verwendung von User-Namespaces zur Verbesserung der Container-Sicherheit
5. Verwendung von AppArmor oder SELinux zur Erzwingung von Richtlinien für Container
6. Verwendung von Secrets zur sicheren Speicherung von vertraulichen Informationen in Containern
7. Verwendung von Security-Scanning-Tools zur Überprüfung von Docker-Images auf Sicherheitslücken
8. Verwendung von Security-Plugins wie Docker Bench oder Clair zur Verbesserung der Sicherheit
9. Verwendung von Audit-Logs zur Überwachung von Container-Aktivitäten
10. Sichere Bereitstellung von Docker in Produktionsumgebungen: Best Practices und Sicherheitsvorkehrungen

Docker-Orchestrierung:

1. Einführung in Docker Swarm: Erstellung und Verwaltung eines Swarm-Clusters
2. Verwendung von Swarm-Services zur Bereitstellung von Anwendungen in einem Swarm-Cluster
3. Skalierung von Swarm-Services: horizontale und vertikale Skalierung
4. Load-Balancing von Swarm-Services mit eingebauten Mechanismen oder externen Load-Balancern
5. Verwendung von Secrets und Configs zur sicheren Konfiguration von Swarm-Services
6. Verwendung von Rolling-Updates zur Aktualisierung von Swarm-Services ohne Ausfallzeiten
7. Verwendung von Healthchecks zur Überwachung der Container-Gesundheit in einem Swarm-Cluster
8. Verwendung von Swarm-Ressourcenlimits zur Ressourcenverwaltung in einem Cluster
9. Verwendung von Swarm-Stacks zur Definition und Verwaltung komplexer Anwendungen
10. Überwachung und Skalierung von Docker Swarm-Clustern

Kubernetes und Docker:

1. Einführung in Kubernetes: Grundlagen und Architektur
2. Installation und Konfiguration eines Kubernetes-Clusters
3. Bereitstellung von Docker-Anwendungen in Kubernetes-Clustern
4. Verwendung von Pods zur Definition von Anwendungsbereitstellung und Skalierung
5. Verwendung von Deployments zur Aktualisierung und Rollback von Anwendungen in Kubernetes
6. Verwendung von Services zur Exposition von Anwendungen in Kubernetes
7. Verwendung von Ingress-Ressourcen zur Konfiguration von HTTP-Routen in Kubernetes
8. Verwendung von ConfigMaps zur Trennung von Konfigurationsdaten und Anwendungscode
9. Verwendung von Secrets zur sicheren Speicherung vertraulicher Informationen in Kubernetes
10. Überwachung und Skalierung von Kubernetes-Anwendungen

Fortgeschrittene Docker-Themen:

1. Verwendung von Docker in einer Microservices-Architektur
2. Verwendung von Docker in einer serverlosen Architektur
3. Integration von Docker mit Container-Orchestrierungsplattformen wie AWS ECS oder GKE
4. Verwendung von Docker in DevOps-Prozessen: CI/CD-Pipelines und automatisierte Tests
5. Verwendung von Docker in einer Big-Data-Umgebung: Hadoop, Spark und andere Tools
6. Verwendung von Docker für Machine Learning und KI-Anwendungen
7. Verwendung von Docker zur Containerisierung von Legacy-Anwendungen
8. Verwendung von Docker zur Entwicklung von IoT-Anwendungen
9. Verwendung von Docker zur Entwicklung von Blockchain-Anwendungen
10. Verwendung von Docker in hochverfügbaren und skalierbaren Architekturen