CodeWizard

visitors 42055 nirgeier

NIRG@CODEWIZARD.CO.IL / 054 8122310

Docker

	Level	Duration
	Introduction	3 days

Course Description

- Docker is a minimal Linux sandbox environment, an open-source engine which automates the deployment of applications as highly portable, self-sufficient containers which are independent of hardware, language, framework, packaging system and hosting provider.

- Docker is a minimal Linux sandbox environment, an open-source engine which automates the deployment of applications as highly portable, self-sufficient containers which are independent of hardware, language, framework, packaging system and hosting provider.

- Docker containers are designed to run isolated services or applications with the ability to share OS, resources, data and more

Audience and prerequisites

- This course is mainly for `Developers` & `DevOps` engineers

Main Goals

- Learn the concept and architecture of Docker
- How to install & use Docker
- Upload self-made container to public registry
- What are images / container / Docker files
- Create and execute Docker containers
- What are Docker files
- How to use existing container
- How to create and publish containers (Hub)
- What is the difference between Docker and virtual machines
- Docker network
- Docker compose

Content

Session	**Content**
Introduction	
	🔹 What is Docker
	🔹 Docker architecture
	🔹 Why Docker is better than other VM's
	🔹 Docker use cases
	🔹 Docker limitations
Architecture	
	🔹 What is LXC
	🔹 Cgroup
	🔹 Union File system (AUFS)
	🔹 Layers
	🔹 Kernel Namespaces
	🔹 Controlled OS resources
	🔹 Docker images
	🔹 Docker containers
	🔹 Repositories
Images & containers	
	🔹 What are images
	The difference between images & containers
	🔹 Images
	🔸 What are images
	🔸 Building images
	🔸 Using images
	🔸 Push/Pull images from Docker hub
	🔹 Containers
	🔸 What are containers
	🔸 Container volumes / Persistence
	🔸 Docker registry
	🔸 Container lifecycle
Docker hub	

	◆ What is a docker hub
	◆ Pull/push images
	◆ Build and tag images
Docker CLI	
	◆ build
	◆ run
	◆ Background / detached
	◆ In foreground
	◆ Interactive
	◆ Ports
	◆ Volumes
	◆ commit
	◆ cp
	◆ diff
	◆ exec
	◆ history
	◆ inspect
	◆ log
	◆ login/logout
	◆ network
	◆ ps
	◆ pull
	◆ push
	◆ search
	◆ stats
	◆ stop / start / pause / resume
	◆ tag
	◆ top
Containers	
	◆ Container lifecycle
	◆ Advanced Docker runs with multiple options
	◆ Volumes

	🔹 Networking
	🔹 Container names
	🔹 Container composition
Docker File	
	🔹 Learn docker-file commands
	🔸 FROM
	🔸 RUN
	🔸 CMD
	🔸 EXPOSE
	🔸 ENV
	🔸 ADD /COPY
	🔸 VOLUME
	🔸 ENTRYPOINT
	🔸 WORKDIR
	🔸 EXPOSE
Docker-compose	
	🔹 In-depth Docker-compose hands-on
	🔸 Networking
	🔸 Isolation
	🔸 Aliases
	🔸 Links
	🔸 Scaling
	🔹 .env files
	🔹 Docker compose commands
	🔸 build
	🔸 config
	🔸 cp
	🔸 down
	🔸 events
	🔸 up
	🔸 wait
	🔸 more ...

	Building multi-container applications using Docker Compose.
Docker Networking	
	🔹 Network in nutshell
	🔸 NAT
	🔸 IP tables
	🔸 routing
	🔹 CNI
	🔹 Network architecture
	🔹 Network drivers
	🔸 bridge
	🔸 host
	🔸 overlay
	🔸 ipvlan
	🔸 macvlan
	🔸 none
	🔹 iptables
	🔸 nat tables
	🔹 Inspect docker network
	🔸 ifconfig
	🔸 brctl
	🔸 iptables
	🔸 NAT
	🔹 Virtual Network
Docker & K8S	
	🔹 Docker & K8S hands-on
	🔸 Convert Docker-compose to K8S resources
	🔸 Containers vs Pods
	🔸 Docker commands in K8S cluster
	🔸 events
	🔸 logs
CI/CD With Docker	
	CI/CD hands-on for building, tagging and pushing docker images to registry

Best practices	
	🔹 .dockerignore
	🔹 caching
	🔹 debugging
	🔹 Docker file HEALTHCHECK
	🔹 docker trust sign
	🔹 MultiStage docker files
	🔹 Optimization + Docker file optimization
	🔹 Pin base images
	🔹 Portainer
	🔹 Security / scan / scout
	🔹 buildx
	🔸 building for different platforms

Back to courses list
