Module Checklist

Containers with Docker

By Techworld with Nana

Video Overview

- ★ What is a Container?
- ★ Container vs Image
- ★ Docker (Container) vs Virtual Machine
- ★ Docker Architecture and its components
- ★ Main Docker Commands
- ★ Debug Commands
- ★ Demo Project: Overview
- ★ Demo Project: Developing with Docker
- ★ Demo Project: Docker Compose - Running multiple services
- ★ Demo Project: Dockerfile - Building our own Docker Image
- ★ Demo Project: Private Docker Repository - Pushing our Docker Image into a private Registry on AWS
- ★ Demo Project: Deploying our containerized application
- ★ Docker Volumes - Persist data in Docker
- ★ Demo Project: Volumes - Configuring persistence for our application
- ★ Docker & Nexus: Push/Pull to Nexus Repository
- ★ Docker & Nexus: Run Nexus as Docker container

Demo Infos	
Git Project	https://gitlab.com/nanuchi/developing-with-docker

Check your progress... 1/6

What is a Container?

- ❏ Watched video

Container vs Image

- ❏ Watched video
- ❏ **Demo executed** - run two different Versions of Postgres Docker Images

Useful Links:

- Postgres Docker Images: https://hub.docker.com/_/postgres

Docker vs Virtual Machine

- ❏ Watched video

Docker components

- ❏ Watched video
- ❏ Installed Docker on your local machine

Useful Links:

- Docker Installation Guides for different OS: https://docs.docker.com/get-docker/

Check your progress... 2/6

Main Docker Commands

- ❑ Watched video
- ❑ **Demo executed**
  - ❑ Pull Redis Docker Image (docker pull)
  - ❑ List existing Docker Images (docker images)
  - ❑ Run Container (docker run)
  - ❑ Run Container in a detached mode (docker run -d)
  - ❑ List running containers (docker ps)
  - ❑ Start container (docker start)
  - ❑ Stop container (docker stop)
  - ❑ List all containers - running and stopped ones (docker ps -a)
  - ❑ Bind port (docker run -p)

Useful Links:

- Redis Docker Images: https://hub.docker.com/_/redis

Debug Commands

- ❑ Watched video
- ❑ **Demo executed**
  - ❑ See logs of container (docker logs)
  - ❑ Get interactive terminal of running container for troubleshooting (docker exec -it)

Demo Project: Overview

- ❑ Watched video

Check your progress... 3/6

Demo Project: Developing with Docker

- ❏ Watched video
- ❏ **Demo executed**
  - ❏ Git cloned example git project or created new one
  - ❏ Pulled mongodb image
  - ❏ Pulled mongo-express image
  - ❏ Created mongo-network
  - ❏ Started mongodb container with all necessary parameters
  - ❏ Started mongo-express container with all necessary parameters
  - ❏ Created new database via Mongo Express UI
  - ❏ Configured Nodejs application code to connect with database

Useful Links:

- MongoDB Docker Image: https://hub.docker.com/_/mongo
- Mongo Express Docker Image: https://hub.docker.com/_/mongo-express
- Demo project: https://gitlab.com/nanuchi/developing-with-docker

Demo Project: Docker Compose - Running multiple services

- ❏ Watched video
- ❏ **Demo executed**
  - ❏ Installed Docker Compose (should already be installed with Docker Desktop)
  - ❏ Created a docker-compose file to start mongodb and mongo-express containers instead of using docker run
  - ❏ Created new database

Useful Links:

- Docker Compose Installation Guides for different OS: https://docs.docker.com/compose/install/

Check your progress... 4/6

Demo Project: Dockerfile - Building our own Docker Image

- ❑ Watched video
- ❑ **Demo executed**
  - ❑ Created Dockerfile for our Node application (the name of the file MUST be Dockerfile!)
  - ❑ Built Docker Image from our Dockerfile and tag it
  - ❑ Started newly created Docker Image

Demo Project: Private Docker Repository - Pushing our Docker Image into a private Registry on AWS

- ❑ Watched video
- ❑ **Demo executed**
  - ❑ Created private Docker Registry on Amazon ECR
  - ❑ Logged in to private registry (docker login)
  - ❑ Tagged Docker Image
  - ❑ Pushed Docker Image to AWS ECR repository

Useful Links:

- Amazon ECR Docker Registry: https://aws.amazon.com/ecr/
- Installing AWS Cli Linux: https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2-linux.html
- Installing AWS CLI on MacOS: https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2-macOS.html
- Installing AWS CLI on Windows: https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2-windows.html
- Configuring the AWS CLI: https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-configure.html

Check your progress... 5/6

Demo Project: Deploying our containerized application

- ❏ Watched video
- ❏ **Demo executed**
  - ❏ Added our example application to Dockerfile
  - ❏ Changed mongodb server url from localhost to mongodb service name in Node Code
  - ❏ Started docker containers with docker-compose

Docker Volumes - Persist data in Docker

- ❏ Watched video

Demo Project: Volumes - Configuring persistence for our application

- ❏ Watched video
- ❏ **Demo executed** - defined a Named Volume in Docker Compose File

Check your progress... 6/6

Docker & Nexus

Push/Pull to Nexus Repository

- ❑ Watched video
- ❑ **Demo executed**
  - ❑ Created a Docker Repository on Nexus
  - ❑ Created a User Role for Docker Repository on Nexus
  - ❑ Configured Repository Connector (port 8083)
  - ❑ Configured Firewall Rule to open port 8083 on Droplet
  - ❑ Configured Token Issuing on Nexus (Realm - activate Docker Bearer Token Realm)
  - ❑ Configured insecure registries for Nexus IP and Port in Docker Desktop (Docker Engine Tab)
  - ❑ Logged in to Nexus Docker Repo (docker login)
  - ❑ Pushed Docker Image to Nexus Repo
  - ❑ Fetched Docker Image from Nexus Repo

Run Nexus as Docker Container on DigitalOcean Droplet

- ❑ Watched video
- ❑ **Demo executed**
  - ❑ Created a new Droplet
  - ❑ Configured Firewall rule to open port 22 for SSHing
  - ❑ Installed Docker on Droplet
  - ❑ Created docker volume to persist Nexus data
  - ❑ Ran Nexus as Docker container with necessary parameters
  - ❑ Accessed Nexus in browser

Useful Links:

- Nexus Docker Image: https://hub.docker.com/r/sonatype/nexus3

More Resources...

Best practices

- Best practices for writing Dockerfiles: https://docs.docker.com/develop/develop-images/dockerfile_best-practices/
- Docker development best practices: https://docs.docker.com/develop/dev-best-practices/
- Tips for Caching, reducing Image size, maintainability, reproducibility: https://www.docker.com/blog/intro-guide-to-dockerfile-best-practices/
- **Security**: Prefer minimal base images (e.g. prefer alpine-based images over full-fledged system OS images)
- **Security**: only use images from trusted vendors to avoid malware
- **Security**: Least privileged user (create a dedicated user and group on the image, with minimal permissions to run the application)
- **Security**: Don't leak sensitive information to Docker Images
- **Tip**: Enforce Dockerfile best practices automatically by using a static code analysis tool (e.g. https://github.com/hadolint/hadolint)