**ΑΚΑΔΗΜΑΪΚΟ ΕΤΟΣ 2024-2025**

**ΕΑΡΙΝΟ ΕΞΑΜΗΝΟ**

**ΜΑΘΗΜΑ: Νεφοϋπολογιστική**

**Παραδείγματα χρήσης docker containers**

1. **Εισαγωγή**

Αυτός ο οδηγός θα σας βοηθήσει να μάθετε τα βασικά του **Docker** και να εκτελέσετε **απλά** και **πιο σύνθετα** παραδείγματα.

1. **Εκτέλεση Βασικών Εντολών Docker**

* **Δοκιμή του Docker**

docker run hello-world

Αυτό θα κατεβάσει και θα εκτελέσει το κοντέινερ hello-world.

* **Εκτέλεση Ubuntu κοντέινερ με διαδραστική (interactive) λειτουργία**

docker run -it ubuntu bash

* + -it: Ενεργοποιεί τη διαδραστική λειτουργία, επιστρέφει δηλαδή στον χρήστη κονσόλα τερματικού στον περιέκτη που ξεκίνησε.
  + Μπορείτε να εκτελέσετε εντολές όπως ls, pwd, exit.
* **Προβολή τρεχόντων κοντέινερ**

docker ps

Για όλα τα κοντέινερ (συμπεριλαμβανομένων των σταματημένων):

docker ps -a

* **Διαγραφή κοντέινερ**

docker rm <container\_id>

* **Διαγραφή εικόνας Docker**

docker rmi <image\_id>

1. **Εκτέλεση Ενός Απλού Web Server με Docker**

* **Εκκίνηση ενός Nginx κοντέινερ**

docker run -d -p 8080:80 nginx

* + -d: Εκτελείται στο παρασκήνιο.
  + -p 8080:80: Χαρτογραφεί τη **θύρα 8080 του host** στη **θύρα 80 του κοντέινερ**.
* **Δοκιμή στο πρόγραμμα περιήγησης**  
  Ανοίξτε το:

http://localhost:8080

Θα δείτε την προεπιλεγμένη σελίδα του Nginx.

1. **Παράδειγμα 1: Δημιουργία νέου Docker κοντέινερ με προσθήκη αρχείων**

Σε αυτό το παράδειγμα, θα φτιάξουμε ένα **custom κοντέινερ** με βάση το **Ubuntu**, το οποίο θα περιέχει ένα απλό **script σε Bash** και θα το εκτελεί κατά την εκκίνηση.

* Δημιουργία φακέλου εργασίας

Αρχικά, δημιουργούμε έναν νέο φάκελο για το project:

**mkdir** **~/**docker-custom-container

**cd** **~/**docker-custom-container

* Δημιουργία αρχείου script.sh

Αυτό το script θα τυπώνει ένα μήνυμα κάθε 5 δευτερόλεπτα.

nano script.sh

Προσθέστε το παρακάτω περιεχόμενο:

#!/bin/bash

**while** **true;** **do**

**echo** "Το Docker κοντέινερ τρέχει! $(date)"

**sleep** 5

**done**

Αποθηκεύστε το (CTRL + X, μετά Y και Enter).

**Δώστε εκτελέσιμα δικαιώματα στο script:**

**chmod** **+**x script.sh

* **Δημιουργία Dockerfile**

Τώρα θα δημιουργήσουμε το **Dockerfile**, το οποίο περιγράφει το κοντέινερ μας.

nano Dockerfile

Επικολλήστε το εξής:

# Χρησιμοποιούμε την εικόνα Ubuntu

FROM ubuntu**:**latest

# Ορίζουμε τον maintainer

LABEL maintainer**=**"example@example.com"

# Ενημέρωση του συστήματος και εγκατάσταση του bash

RUN apt-get update **&&** **apt-get** install -y bash

# Αντιγραφή του script μέσα στο κοντέινερ

COPY script.sh **/**script.sh

# Ορισμός δικαιωμάτων εκτέλεσης στο script

RUN chmod **+**x **/**script.sh

# Εκτέλεση του script κατά την εκκίνηση του κοντέινερ

CMD **[**"/script.sh"**]**

Αποθηκεύστε το αρχείο.

* **Δημιουργία του Docker Image**

Τώρα θα φτιάξουμε την εικόνα Docker:

docker build -t my-custom-container .

* **Εκτέλεση του κοντέινερ**

Εκτελέστε το κοντέινερ στο παρασκήνιο:

docker run -d --name my-container my-custom-container

Για να δείτε τις καταγραφές του κοντέινερ σε πραγματικό χρόνο:

docker logs -f my-container

Θα δείτε μηνύματα όπως:

Το Docker κοντέινερ τρέχει! Tue Mar 5 12:00:00 UTC 2025

Το Docker κοντέινερ τρέχει! Tue Mar 5 12:00:05 UTC 2025

...

* **Διαχείριση και καθαρισμός**

**Σταμάτημα του κοντέινερ:**

docker stop my-container

**Διαγραφή του κοντέινερ:**

docker rm my-container

**Διαγραφή της εικόνας:**

docker rmi my-custom-container

1. **Παράδειγμα 2: Εκτέλεση Σύνθετου Παραδείγματος με Nginx και Πολλαπλούς Web Servers**

Θα δημιουργήσουμε ένα **Docker Compose setup** με:

* **Nginx ως reverse proxy**
* **Δύο web servers** με απλές HTML σελίδες
* Δημιουργία φακέλου εργασίας

**mkdir** **~/**docker-nginx-multi

**cd** **~/**docker-nginx-multi

* Δημιουργία docker-compose.yml

Εκτελέστε:

nano docker-compose.yml

Επικολλήστε το παρακάτω περιεχόμενο:

**version**: "3.8"

**services**:

**web1**:

**image**: nginx

**container\_name**: web1

**volumes**:

- ./web1:/usr/share/nginx/html

**networks**:

- mynetwork

**web2**:

**image**: nginx

**container\_name**: web2

**volumes**:

- ./web2:/usr/share/nginx/html

**networks**:

- mynetwork

**nginx**:

**image**: nginx

**container\_name**: nginx-proxy

**volumes**:

- ./nginx.conf:/etc/nginx/nginx.conf:ro

**ports**:

- "8080:80"

**depends\_on**:

- web1

- web2

**networks**:

- mynetwork

**networks**:

**mynetwork**:

* **Δημιουργία φακέλων για HTML αρχεία**

**mkdir** web1 web2

* **Δημιουργία HTML σελίδων**
  + Για τον **Web1 server**:

**echo** "<h1>Welcome to Web1</h1>" **>** web1**/**index.html

* + Για τον **Web2 server**:

**echo** "<h1>Welcome to Web2</h1>" **>** web2**/**index.html

* **Δημιουργία αρχείου** nginx.conf **(Reverse Proxy)**

Εκτελέστε:

nano nginx.conf

Επικολλήστε το παρακάτω:

events **{}**

http **{**

upstream backend **{**

server web1**;**

server web2**;**

**}**

server **{**

listen 80**;**

location **/** **{**

proxy\_pass http**://**backend**;**

**}**

**}**

**}**

* **Εκκίνηση των κοντέινερ**

docker compose up -d

* **Δοκιμή στο πρόγραμμα περιήγησης**

Ανοίξτε:

http://localhost:8080

Το Nginx θα εναλλάσσει αιτήματα μεταξύ **Web1** και **Web2**.

1. **Docker Volumes: Διαφορά Ephemeral και Persistent Volumes**

Στο Docker, υπάρχουν **δύο τύποι αποθήκευσης δεδομένων**:

* + **Ephemeral (Προσωρινή) Αποθήκευση** – Τα δεδομένα χάνονται όταν το κοντέινερ διαγραφεί.
  + **Persistent (Μόνιμη) Αποθήκευση** – Τα δεδομένα διατηρούνται ανεξάρτητα από το κοντέινερ.

Ας δούμε τη διαφορά με πρακτικά παραδείγματα.

* **Ephemeral Storage (Τα δεδομένα χάνονται)**

Τα δεδομένα αποθηκεύονται μέσα στο σύστημα αρχείων του κοντέινερ και **δεν επιβιώνουν** όταν το κοντέινερ διαγραφεί.

* **Βήμα 1: Εκκίνηση Κοντέινερ και Δημιουργία Αρχείου**

docker run -it --name temp-container ubuntu bash

Μέσα στο κοντέινερ, δημιουργούμε ένα αρχείο:

**echo** "Προσωρινά δεδομένα" **>** **/**tmp**/**tempfile.txt

**cat** **/**tmp**/**tempfile.txt

Θα δείτε:

Προσωρινά δεδομένα

* **Βήμα 2: Διαγραφή Κοντέινερ και Έλεγχος Δεδομένων**

Διαγράφουμε το κοντέινερ:

docker rm temp-container

Ξεκινάμε ένα **νέο κοντέινερ** και ελέγχουμε αν το αρχείο υπάρχει:

docker run -it ubuntu bash

**ls** **/**tmp

Το αρχείο **δεν υπάρχει** γιατί το σύστημα αρχείων του κοντέινερ ήταν προσωρινό (ephemeral).

* **Persistent Storage (Τα δεδομένα διατηρούνται)**

Τα δεδομένα αποθηκεύονται **εκτός του κοντέινερ**, σε ένα Docker **Volume**, και διατηρούνται ακόμα και μετά τη διαγραφή του κοντέινερ.

* **Βήμα 1: Δημιουργία Volume**

docker volume create mydata

Επιβεβαιώνουμε ότι το volume δημιουργήθηκε:

docker volume ls

* **Βήμα 2: Εκκίνηση Κοντέινερ με Volume**

docker run -it --name persistent-container -v mydata:/data ubuntu bash

Μέσα στο κοντέινερ, δημιουργούμε ένα αρχείο:

**echo** "Μόνιμα δεδομένα" **>** **/**data**/**persistentfile.txt

**cat** **/**data**/**persistentfile.txt

Θα δείτε:

Μόνιμα δεδομένα

Βγαίνουμε από το κοντέινερ:

exit

* **Βήμα 3: Διαγραφή Κοντέινερ και Έλεγχος Δεδομένων**

Διαγράφουμε το κοντέινερ:

docker rm persistent-container

Τώρα ξεκινάμε ένα νέο κοντέινερ και ελέγχουμε αν τα δεδομένα υπάρχουν:

docker run -it --rm -v mydata**:/**data ubuntu bash

**ls** **/**data

**cat** **/**data**/**persistentfile.txt

Θα δείτε ότι το αρχείο **υπάρχει ακόμα**:

persistentfile.txt

Μόνιμα δεδομένα