INTERNET PAMETINIH UREĐAJA

prof. dr Dejan S. Aleksić

Prirodno-matematički fakultet, Niš

07.B KORIŠĆENJE DOCKER-A U RAZVOJU IOT APLIKACIJA

CILJEVI PREDAVANJA

Nakon ovog predavanja biće jasnije:

- Šta je Docker i zašto se koristi
- Kako da instalira Docker Desktop i WSL na Windows
- Kako napraviti Go web aplikaciju
- Kako da formirate Docker image
- Kako da prebacite image na Docker Hub
- Kako da pokrenete aplikaciju na Ubuntu serveru (VM)
- Kako da testirate aplikaciju iz Windows-a

PROBLEM KADA SE RADI BEZ DOCKER-A

❌ Različiti operativni sistemi
❌ Različite verzije biblioteka
❌ „Kod radi kod mene, ali ne kod tebe“
❌ Teško premeštanje aplikacije na server

➡️ **Rešenje: Docker**

ŠTA JE DOCKER?

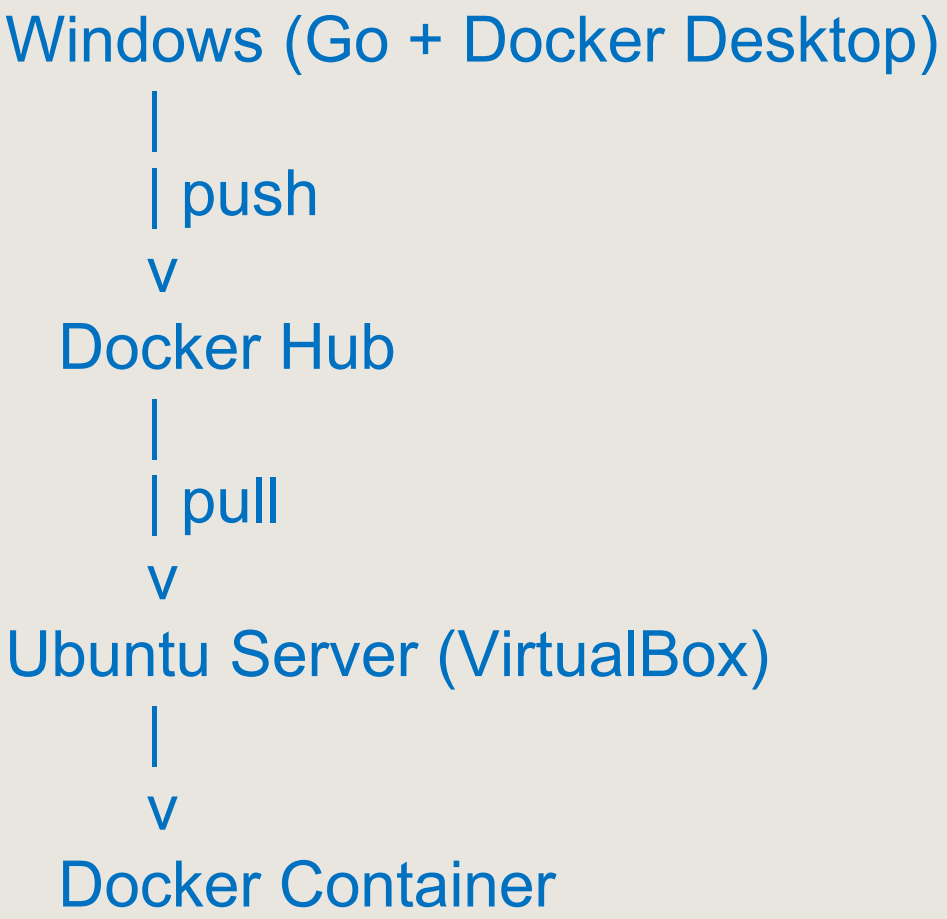
Docker je platforma za:

- Pakovanje aplikacije + zavisnosti
- Izvršavanje aplikacije u **kontejnerima**
- Brz deployment na servere i cloud

📦 **Image** – nepromenljivi šablon
📦 **Container** – pokrenuta instanca image-a

DOCKER U NAŠEM SCENARIJU

1. Razvoj na Windows
2. Docker image sa Go aplikacijom
3. Push na Docker Hub
4. Ubuntu VM = production server
5. Pull image-a
6. Pokretanje kontejnera



ŠTA JE WSL?

WSL (Windows Subsystem for Linux) omogućava:

- Pokretanje Linux-a unutar Windows-a
- Rad Docker-a bez klasične VM
- Brži i stabilniji razvoj

➡️ Docker Desktop koristi **WSL2**

INSTALACIJA WSL-A

U PowerShell (Admin):

```
wsl --install
```

- Restart računara
- Instalira se Ubuntu
- Proveriti:

```
wsl --list --verbose
```

PREDUSLOVI ZA INSTALACIJU

U PowerShell (Admin):

```
wsl --install
```

- Restart računara
- Instalira se Ubuntu
- Proveriti:

```
wsl --list --verbose
```

INSTALACIJA DOCKER DESKTOP-A

Provera virtualizacije:

Pre instalacije, proverite da li je virtualizacija omogućena na vašem procesoru:

1. Otvorite Task Manager (Ctrl+Shift+Esc).
2. Idite na tab Performance -> CPU.
3. Proverite polje Virtualization. Ako je Disabled, morate je omogućiti u BIOS-u.

Instalacija Docker Desktop-a:

1. Preuzeti sa: 👉 https://www.docker.com/products/docker-desktop
2. Instalirati
3. Omogućiti:
   - ✅ WSL 2 backend
   - ✅ Ubuntu distro
4. Test:

```
docker version
docker run hello-world
```

PRVA GO WEB APLIKACIJA

Minimalni HTTP server u Go-u:

```
package main

import (
    "fmt"
    "net/http"
)

func handler(w http.ResponseWriter, r *http.Request) {
    fmt.Fprintln(w, "Hello from Docker + Go!")
}

func main() {
    http.HandleFunc("/", handler)
    http.ListenAndServe(":8080", nil)
}
```

TEST APLIKACIJE LOKALNO

Pokretanje aplikacije pod windows-om:

```
go run main.go
```

Otvoriti u browser-u:

```
http://localhost:8080
```

ŠTA JE DOCKERFILE?

Dockerfile opisuje:

- Koji OS koristimo
- Kako se aplikacija gradi
- Koji port se izlaže
- Kako se aplikacija pokreće

➡️ Automatski build image-a

DOCKERFILE ZA GO APLIKACIJU (MULTI-STAGE BUILD)

Zašto multi-stage build?

- ✓ Manja veličina image-a
- ✓ Bez Go kompajlera u produkciji
- ✓ Brže preuzimanje i veća sigurnost

DOCKERFILE ZA GO APLIKACIJU (MULTI-STAGE BUILD)

```
# ===== BUILD STAGE =====
FROM golang:1.22-alpine AS builder

WORKDIR /app

COPY go.mod ./
COPY main.go ./

RUN go build -o app

# ===== RUNTIME STAGE =====
FROM alpine:latest

WORKDIR /app

COPY --from=builder /app/app .

EXPOSE 8080

CMD ["./app"]
```

Šta se ovde dešava?

1 builder faza
- Koristi se Go image
- Aplikacija se kompajlira u binarni fajl

2 runtime faza
- Koristi se minimalni Alpine Linux
- Kopira se samo binarni fajl
- Image je mali i bez nepotrebnih alata

Prednost u praksi
- Image veličina: ~10–15 MB
- Idealno za production servere i IoT okruženja

BUILD DOCKER IMAGE-A

```
docker build -t username/go-web-app .
```

Provera:

```
docker images
```

Pokretanje kontejnera lokalno:

```
docker run -p 8080:8080 username/go-web-app
```

Test:

```
http://localhost:8080
```

UPRAVLJANJE PORTOVIMA (-P FLAG)

- **Host vs Container:** Kontejner ima svoju izolovanu IP adresu i portove.
- **Mapiranje:** -p 8080:80
  - 8080: Port na vašem Windows laptopu.
  - 80: Port unutar Docker kontejnera gde Go server sluša.
- **Primer:** Ako pokrenete dva kontejnera, oba unutra mogu slušati na 80, ali ih napolju mapirate na 8081, 8082 itd.

Test:

```
docker run -p 8080:8080 username/go-web-app
```

```
http://localhost:8080
```

```
docker run -p 8080:80 username/go-web-app
```

```
http://localhost:80
```

DOCKER HUB

Docker Hub je:

- Online registry za Docker image-e
- Omogućava deljenje i deployment
- Napravite besplatan nalog na hub.docker.com.

SLANJE IMAGE-A NA DOCKER HUB

Naming convention: `korisnicko_ime/ime_projekta:verzija`
Zašto verzije? U IoT-u je bitno imati "Rollback". Ako verzija v2 padne, lako povlačimo v1.

1. Prijava: (uneti username i password)

```
docker login
```

2. Tagovanje (ako već niste):

```
docker tag go-web-app username/go-web-app:v1
```

3. Slanje:

```
docker push username/go-web-app:V1
```

UBUNTU SERVER (VIRTUALBOX)

Ubuntu VM simulira:

- Produkcioni server
- IoT backend node

Instalirano:

- Ubuntu Server
- Docker Engine

INSTALACIJA DOCKER-A NA UBUNTU

```
sudo apt update
sudo apt install docker.io
sudo systemctl start docker
sudo systemctl enable docker
```

Test:

```
docker run hello-world
```

PREUZIMANJE I POKRETANJE APP NA UBUNTU

Pull image-a sa Docker Hub-a

```
docker pull username/go-web-app
```

Pokretanje aplikacije na serveru

```
docker run -d -p 8080:8080 username/go-web-app:v1
```

Testiranje:

U browser-u na Windows-u:

http://IP_ADRESA_VM:8080

➡️ Aplikacija radi u Docker kontejneru na Ubuntu serveru

PROVERA I BRISANJE RESURSA

Nakon što ste uspešno poslali sliku na Docker Hub, obrišite lokalni kontejner i sliku kako biste oslobodili prostor:

```
# Zaustavljanje i brisanje kontejnera
docker stop username/go-web-app
docker rm username/go-web-app

# Brisanje lokalne slike
docker rmi go-web-app
```

Korisne docker komande:

- `docker ps` - Lista pokrenutih kontejnera.
- `docker images` - Lista svih lokalnih imidža.
- `docker logs <ime>` - Pregled logova aplikacije (šta ispisuje fmt.Println).
- `docker stop $(docker ps -q)` - Zaustavljanje svih kontejnera.

ZAŠTO JE OVO VAŽNO ZA IOT?

- ✓ Standardizovan deployment
- ✓ Lak prenos sa razvoja na produkciju
- ✓ Skalabilnost
- ✓ Idealno za mikroservise
- ✓ Cloud & Edge podrška

https://labs.play-with-docker.com/