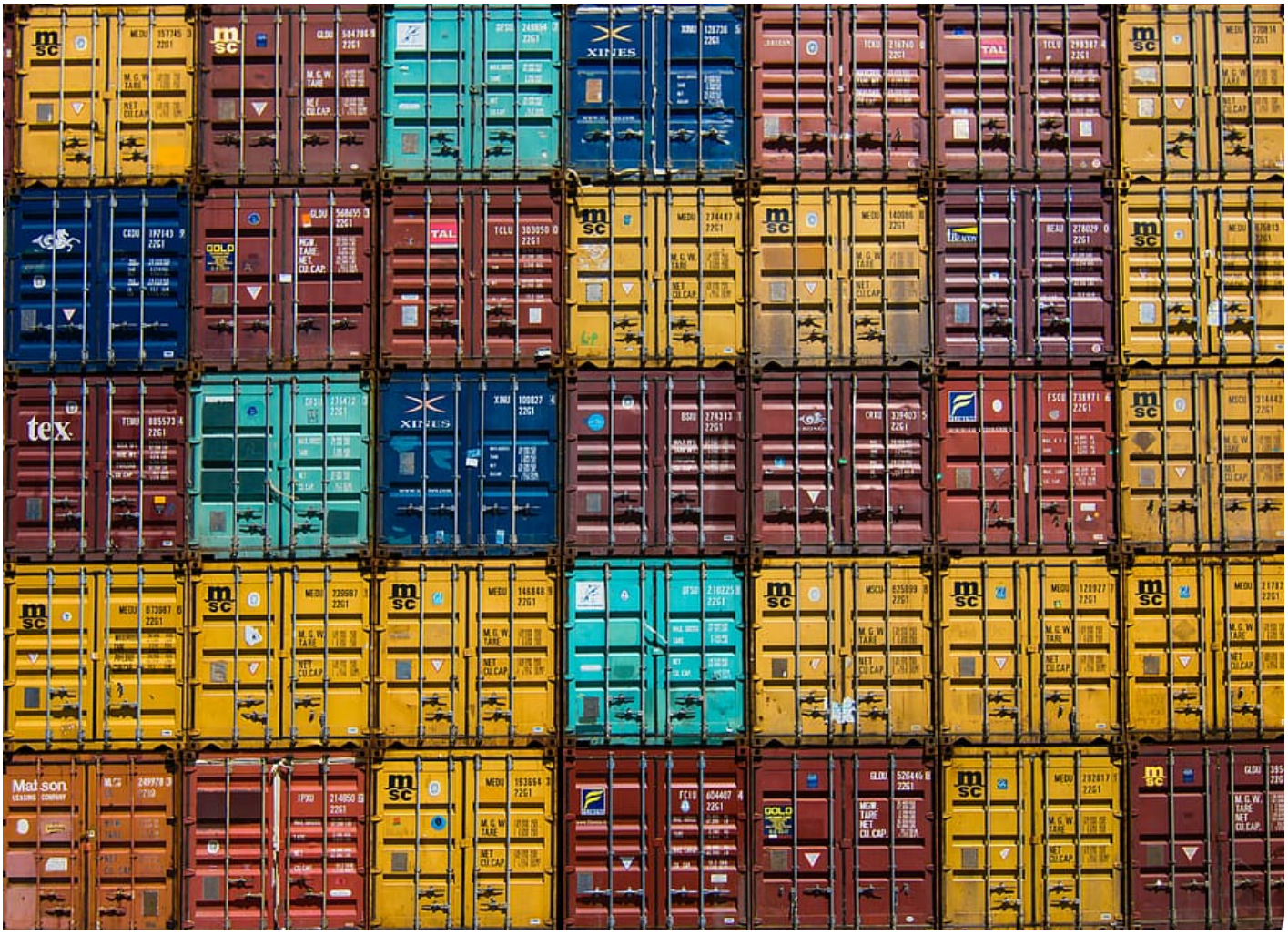
docker-compose.yml – set instrukcija za stvaraje kontejnera



Docker compose možete posmatrati kao set instrukcija koji će dati vašem imidžuu neki dodatni set funkcionalnosti a koji ne mogu da se definišu unutar Dockerfajla. Uz njega možete izbeći sve one komplikacije gde morate u dugaćkim kobasicama navoditi šta želite da vaš kontejner ima otvoreno od portova, koje mount pointe da koristi, kako da mu bude ime definisano.

Neki od vas su sigurno bar jednom otkuali `docker-compose up`, možda baš u laradocku, i naveli šta sve žele da se startuje od usluga. E ovde ću pokušati da vam prikažem kroz objašnjenje jednog

nasumičnog yaml fajla sa interneta, šta sve docker-compose može.

Podizanje wordpressa

```
Version: '3.3'

services:
   db:
     image: mysql:5.7
     volumes:
       - db_data:/var/lib/mysql
     restart: always
     environment:
       MYSQL_ROOT_PASSWORD: somewordpress
       MYSQL_DATABASE: wordpress
       MYSQL_USER: wordpress
       MYSQL_PASSWORD: wordpress

   wordpress:
     depends_on:
       - db
     image: wordpress:latest
     ports:
       - "8000:80"
     restart: always
     environment:
       WORDPRESS_DB_HOST: db:3306
       WORDPRESS_DB_USER: wordpress
       WORDPRESS_DB_PASSWORD: wordpress
       WORDPRESS_DB_NAME: wordpress
volumes:
    db_data: {}
```

NAPOMENA: Kod yaml kodiranja, jako su bitni indenti. Ako tu omašite za jedan razmak, yaml fajl neće raditi.

Komentar je isti kao u većini programskih jezika, počinje sa znakom # . Sada ću kopirati kod od

iznad, i iskomentarisati vam u kodu i inline šta znači šta.

```
Version: '3.3' #Verzija zavisi od toga koji docker engine koristite. Može se desiti da sintaksa koju koristite ne bude podržana na starijim enginima.

services: #Ovde navodimo koje servise želimo da podignemo i definišemo imena
   db: #Definišemo ime servisa koji će gurati bazu podataka
     image: mysql:5.7 #Ovde navodimo od kog imidža sa docker huba se ovaj kontejner/servis spawnuje
     volumes: #ovde definišemo persistentnu memoriju kontejnera.
       - db_data:/var/lib/mysql # lokalni_dir:/lokacija/u/kontejneru
     restart: always # U slučaju da nešto krene naopako pokrećemo ga opet
     environment: #Definišemo globalne promenljive ovog kontejnera
       MYSQL_ROOT_PASSWORD: somewordpress
       MYSQL_DATABASE: wordpress
       MYSQL_USER: wordpress
       MYSQL_PASSWORD: wordpress

   wordpress: #Ime jednog od servisa, obratite pažnju da je u istom indentu kao i db gore
     depends_on: #Ovde možemo da kažemo ovom kontejneru da čeka dok se ne podigne neki kontejner
       - db  #U stvari čekamo bazu, ili php ima da kaže da ne vidi bazu, i srušiće se kontejner
     image: wordpress:latest #Opet navodimo koji image želimo da podginemo
     ports: #Definišemo koje portove želimo da prosledimo lokalnoj mašini sa dockera
       - "8000:80"
     restart: always #Isti uslov kao i za bazu
     environment:
       WORDPRESS_DB_HOST: db:3306 #Obratite pažnju, ovde gađamo ime servisa:kroz port u docker networku
       WORDPRESS_DB_USER: wordpress
       WORDPRESS_DB_PASSWORD: wordpress
       WORDPRESS_DB_NAME: wordpress
volumes:
    db_data: {} #Ovde mountujemo db_data iz meni nepoznatog razloga u okviru wordpressa :)
```

Ovaj fajl sačuvajte kao docker-compose.yml i napravie direktoriju db_data. Nakon što ispalite dokcer-compose up. DIžu se se DVA kontejnera/servisa. Kontejner koji se zove db i kontejner koji

se zoba wordpress. I oni komuniciraju bezbedno kroz docker mrežu. Ali to je već priča za sebe. O tome drugi put.

Ako budete imali pitanja, postavite ih u komentaru na fejsbuk stranici. A ja ću dopisati odgovore u okviru ovog članka.

Revision #1
Created 10 July 2021 14:25:19 by Admin
Updated 10 July 2021 14:25:36 by Admin