Docker

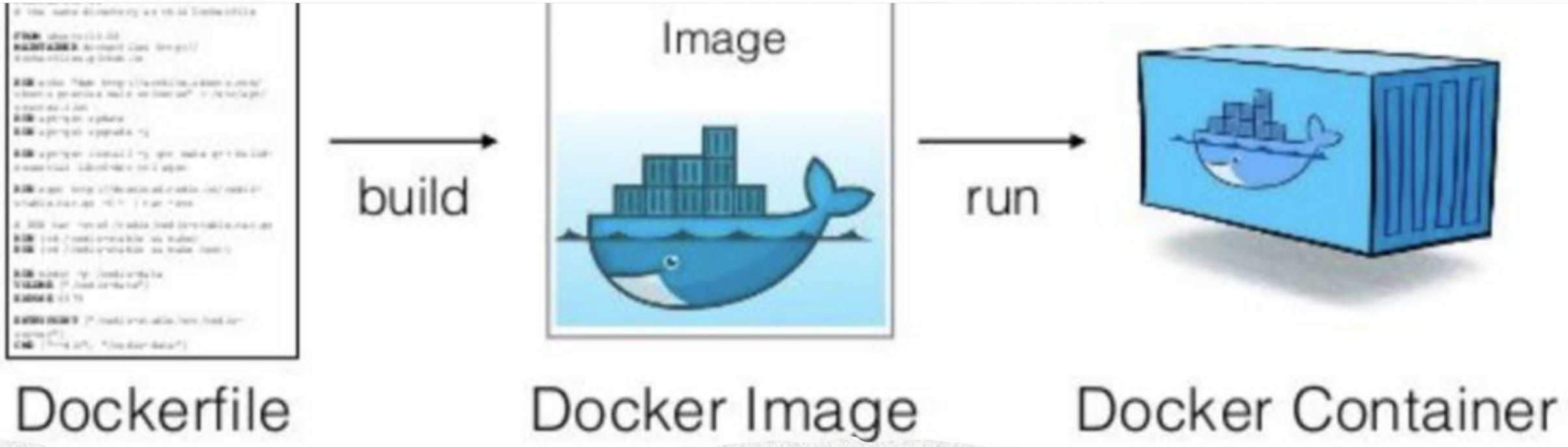
Şeyma Berber

Docker Nedir?

- Docker, uygulama geliştirmek, dağıtmak ve çalıştırmak için oluşturulan bir platformdur.
- 2013 senesinde ortaya çıktı.
- Açık kaynaklıdır.

Neden Docker Kullanılır?

- **1. Taşınabilirlik (Portability)**
- “Benim makinemde çalışıyor ama seninkinde çalışmıyor” sorununu ortadan kaldırır.
- **2. Kaynak Verimliliği**
- Geleneksel sanal makinelerden daha az kaynak kullanır.
- **3. Hızlı ve Kolay Dağıtım**
- Docker image’ları hızlıca build edilir ve dağıtılır.
- **4. Tutarlılık (Consistency)**
- Sürpriz hata olasılığı azalır.
- **5. Kolay Geri Alma (Rollback)**
- Versiyonlanmış Docker image’ları sayesinde uygulamalar istenilen versiyona hızla döndürülebilir.
- **6. Mikroservis Mimarisine Uygunluk**
- Her servis bağımsız bir container içinde çalışabilir.
- Servisler birbirinden izole edilir ve kolayca yönetilir.
- **7. Topluluk Desteği ve Hazır Image’lar**
- Docker Hub’da binlerce hazır image bulunur.



Bir uygulamamızı Docker bünyesinde ayağa kaldırabilmek(dockerize yapmak) için yukarıdaki adımları gerçekleştirmemiz gerekiyor.

Dockerfile

Bu dosya içerisine talimatlar yazarız. Bu talimatlar uygulamamızın nasıl bir ortamda çalışacağı ile ilgili talimatlardır.

Uzantısı yoktur.

Bir uygulamanın ve o uygulamanın çalışması için gerekli tüm ek kütüphane ve diğer öğelerin paketlenmiş haline denir.

IMAGE

Image, Dockerfile içeriği doğrultusunda oluşturulur.

Layer topluluğudur.

Static bir dosyadır. Yani sadece okunabilir. Çalıştırılamaz

Docker image isimlendirme yapısı

Docker imagelerine verilen isim o imajın depolandığı yeri de belirtir.

Her objenin benzersiz bir ID'si vardır ve Docker objeler bu ID'ler üzerinden tanınır.

Docker Image isimleri üç temel kısımdan oluşur.
1- Registry URL : Imagenin durduğu ya da duracağı registry url'i belirtir.
2- Repository: Imagenin ne olduğunu ve kimin tarafından oluşturulduğunu belirtir.
3- Tag: Versionu belirtir. Aynı repositoryden birden fazla imageye ulaşmayı sağlar. Biz tag vermezsek Docker varsayılan olarak latest tagını verir.
Örn: docker.io/ubuntu:18.04

Container

Docker image’lerin çalışabilir instance’leridir.

Başlatabilir, durdurabilir, silebiliriz.

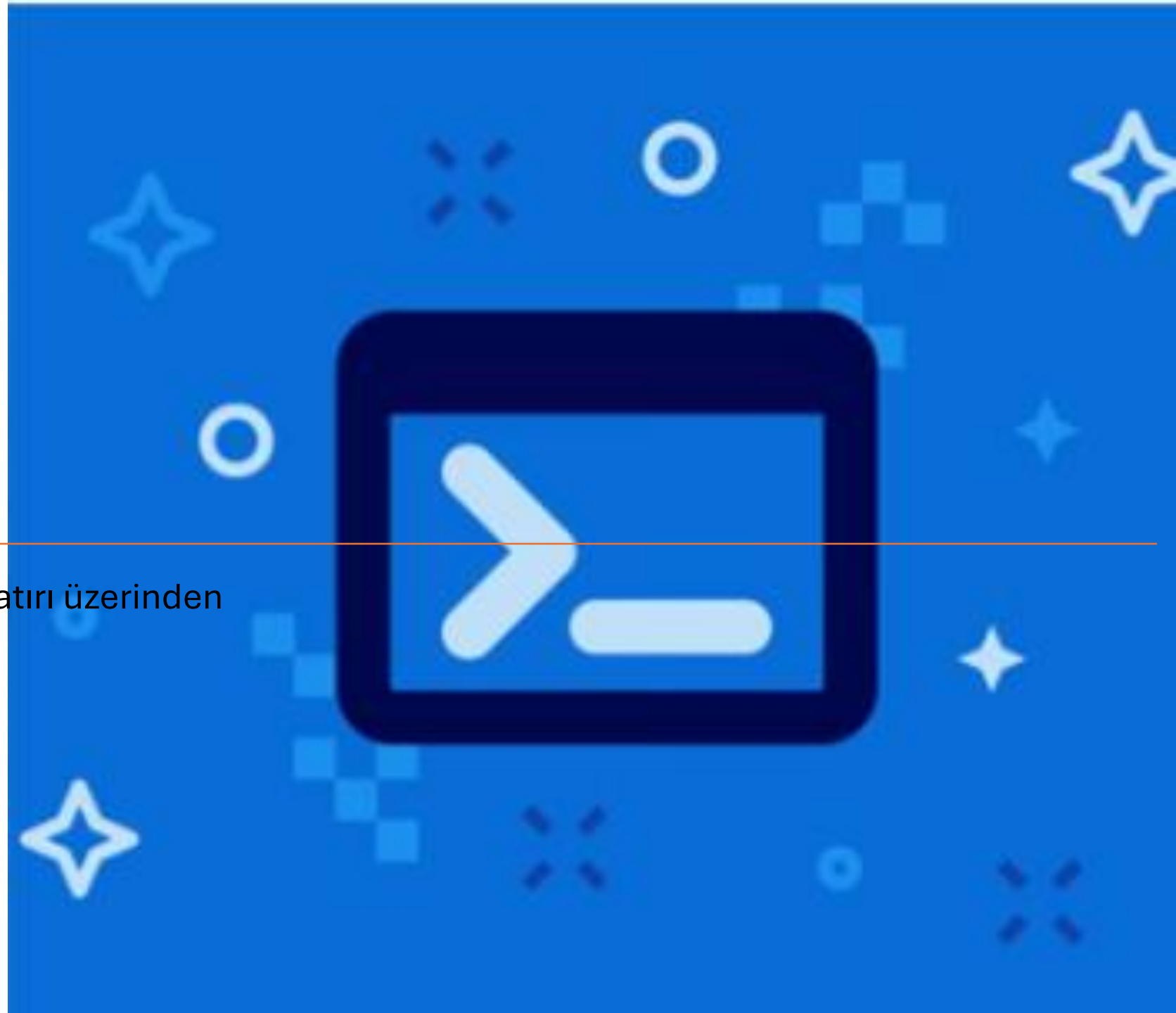
Docker Image üzerinden istediğimiz kadar container ayağa kaldırabiliriz.

Hangi sisteme hangi Docker kurulacak?

- Linux → Docker Engine CE
- Mac OsX → Docker Desktop for Mac
- Windows 10 Pro-Ent-edu → Docker Desktop for Windows
- Windows 7-8 ve 10 Home → Docker Toolbox

Docker CLI

Docker’ı terminal veya komut satırı üzerinden yönetmenizi sağlayan araçtır.

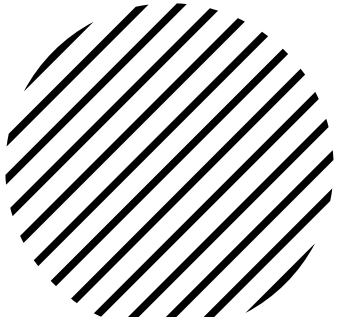


Volume

- Conteynerlar uzun ömürlü değildir. Fakat verilerin uzun ömürlü olması gerekir. Bu sorunu çözmek için volume geliştirilmiştir.

- Docker konteynırlar tarafından üretilen dataları kalıcı kılmamızı sağlayan bir yöntemdir. Container silinsede volume silinmez.

- Bir volumeyi birden fazla container’a bağlayabiliriz.



Environment Variables

Environment variables, işletim sisteminin tamamında geçerli olan ve her yerden çağırılabilen isimlerdir. Yani işletim sistemi bazında tanımlanır.

! Environment oluştururken büyük küçük harfe dikkat etmeliyiz. Çünkü var1 ile VAR1 aynı şeyler değildir.

Docker Registry

Docker imajlarının saklandığı, yönetildiği ve paylaşıldığı merkezi bir sistemdir. Yani, bir tür image deposudur.

Uygulamanızın Docker imajını bir registry'e gönderebilir (push), sonra başka bir yerde bu imajı çekip (pull) kullanabiliriz.

Bazı Docker Reistry'ler

- Docker Hub
- GitHub Container Registry (GHCR)
- Amazon Elastic Container Registry (ECR)
- GitLab Container Registry
- Google Artifact Registry
- Azure Container Registry (ACR) ...

Docker compose?

Docker Compose, docker-compose.yml adında bir dosya içinde, birden fazla servisi (container'ı) tanımlamamıza olanak tanır. Ardından sadece bir komutla tüm servisleri başlatabilir, durdurabilir veya yeniden oluşturabiliriz.

Neden	Açıklama
Kolay çoklu servis yönetimi	Tek bir dosya ve komutla tüm servisleri yönetebilme
Ortam taşınabilirliği	Her ortamda aynı şekilde çalışabilir sistem
Otomatik network & bağlanabilirlik	Servislerin isimleriyle otomatik haberleşme
Geliştirici deneyimi	Hızlı, pratik ve basit geliştirme deneyimi
Volume, env, network yönetimi	Merkezi yapılandırma, volume ve network yönetimi
CI/CD uyumu	Test ve dağıtım süreçlerine kolay entegrasyon

Neden Docker Compose Kullanılır?