Merhabalar önceki docker yazımda (TIKLAYINIZ) Docker kurulumunu görmüştük şimdi ise basit komutlara geçelim … Terminalimizi açalım ve aşağıdaki kodları deneyerek ilerleyelim.

NOT : Docker 17.12.1-ce sürümünde bu kodları yazmaktayım.

Docker ile hangi komutları kullanabileceğimizi ve açıklamalarını gösteren komuttu yazıp enter tuşuna basalım …

```
docker --help
```

peki bir komutu seçtik diyelim(Örneğin : **stop**) peki bunu nasıl kullanacağız öğrenmek için aşağıdaki komutu girelim ;

```
docker stop --help
```

Docker versiyonunu basitçe öğrenmek için ;

```
docker -v
```

Daha geniş kapsamlı docker versiyonunu öğrenmek için

```
docker version
```

NOT : Bu kodu yazdığınızda GO versiyonunu göreceksiniz Docker GO dili ile yazılmıştır …

Aşağıdaki kodu yazdığınızda ise İşletim Sisteminiz ile ilgili pekçok veriye erişeceksiniz.

```
sudo docker info
```

Ufak da olsa bir giriş yaptık.Biraz tanımlama yapalım …

Docker ‘da Container Nedir ?

Container bir process in (işlemin) tek başına tek iş yapacak şekilde diğer çalışan işlemlerden izole bir şekilde çalıştırılabilmesi işlemidir. Bu amacı taşıyan bir tanımdır.Dallanıp budaklandırılabilir ne gibi peki tek başına birçok iş yapabilir izole ortamı kendimiz belirleyerek bozabiliriz ancak amacı aşmış oluruz.Container ların amacı tek bir container da tek bir işi yapabilmesi durumudur.Bu sebep ile her yaratılan container ‘ın tek bir görevi olmalıdır.

Docker Image Nedir ?

Amaca yönelik container ların çalışması için gerekli önceden tasarlanmış kalıplardır.Bir nevi tak çalıştır işletim sistemi gibi … Bu kalıpları ister kendimiz yaratabilir ister isek docker hub(eski) ,docker store(yeni) veya kendi registry ımızdan temin edebiliriz hatta taşınabilir belleklere aktarıp taşıyabiliriz.Bu arada registry nedir ? Docker için registry Image larımızı tutulduğu depodur.İstediğimiz zaman burada image larımızı saklayabilir istediğimiz zaman buradan temin edebiliriz.

Docker ın Resmi Registry si ;

Docker Imagelerinin bulunduğu yeni registry;

https://store.docker.com/

Docker Imagelarının bulunduğu eski registry;

https://hub.docker.com/

Genele açık olan imageları bu sitelerden temin edebiliriz.Bir nevi github da ki

kaynak kodunu indiriyormuş gibi bize gerekli imageları buradan indirebiliriz.Buradaki Docker Image Store larda image ı nasıl bilgisayara indirebileceğimiz yazmaktadır.Örneğin wordpress image ını indirmek isteyelim bunun için yazmamız gereken kod ;

```
docker pull wordpress
```

Sonrasında image'ımız inmeye başlayacaktır.İndikten sonra kullanıma hazır Nasıl kullanacağımızı sonra geleceğiz.Şimdilik bu genele açık olan image ların güvenilir olup olmadığını nereden anlayabiliriz ? Belirttiğim sitelerde official yazan image lar veya başında bir isimlendirme gelmeyen imagelar("docker pull ubuntu" komutundaki ubuntu gibi) güvenilir.Diğerleri ise(örn : docker pull serkankaya/ubuntu) kullanıcıların kendilerine göre imageları düzenleyip attıkları imagelar.Peki bu imagelar güvenilir mi ? Bunun için biraz araştırma yapmamız gerekmekte.Öncelikle bu imageların güvenilir olup olmadığını görmek için Dockerfile(Sonra açıklayacağım) dosyası olup olmadığına sonra bu dosyayı inceleyerek kök image'a kadar olan yolda izini sürmemiz gerekir.Ancak bu şekilde güvenilir bir image olduğuna karar verebiliriz. Yok bu yol bana göre değil genele açık olan registry 'ı kullanmam kendi registry'ımı kurmak kendi image ımı yaratmak istiyorum derseniz Docker 'ın bu noktada da çözümleri bulunmakta.Bu konulara da sonra değineceğim.

Şimdi aşağıdaki komut ile Docker resmi registry den nasıl image indirebileceğimizi gördük.

```
docker pull &lt;IndirilecekImageAdi&gt;
```

Peki her seferinde bu imagelara docker ın sitesinden mi bakmak zorundayız tabiki hayır. Docker registry üzerinden image aramak için gerekli komut aşağıdaki gibi ;

```
docker search &lt;AranacakImageAdi&gt;
```

Aranacak image adı yerine istediğiniz değeri girin bir karşılığı var ise isimleri ile birlikte listelenecektir.Örneğin “docker search ubuntu” girdim terminalden gelen çıktı aşağıdaki gibi ;

```
NAME                                                      DESCRIPTION
STARS               OFFICIAL            AUTOMATED
ubuntu                                                    Ubuntu is a Debian-
based Linux operating sys…   7340                [OK]
dorowu/ubuntu-desktop-lxde-vnc                            Ubuntu with openssh-
server and NoVNC            167                                   [OK]
rastasheep/ubuntu-sshd                                    Dockerized SSH
service, built on top of offi…   132
[OK]
ansible/ubuntu14.04-ansible                               Ubuntu 14.04 LTS
with ansible                    90                                  [OK]
ubuntu-upstart                                            Upstart is an event-
based replacement for th…   81                  [OK]
neurodebian                                               NeuroDebian provides
neuroscience research s…   46                   [OK]
ubuntu-debootstrap                                        debootstrap --
variant=minbase --components=m…   35                    [OK]
1and1internet/ubuntu-16-nginx-php-phpmyadmin-mysql-5      ubuntu-16-nginx-php-
phpmyadmin-mysql-5          27                                    [OK]
nuagebec/ubuntu                                           Simple always
updated Ubuntu docker images w…   22
[OK]
tutum/ubuntu                                              Simple Ubuntu docker
images with SSH access     18
ppc64le/ubuntu                                            Ubuntu is a Debian-
based Linux operating sys…   11
i386/ubuntu                                               Ubuntu is a Debian-
based Linux operating sys…   10
1and1internet/ubuntu-16-apache-php-7.0                    ubuntu-16-apache-
php-7.0                         7                                   [OK]
1and1internet/ubuntu-16-apache-php-5.6                    ubuntu-16-apache-
php-5.6                         6                                   [OK]
eclipse/ubuntu_jdk8                                       Ubuntu, JDK8, Maven
3, git, curl, nmap, mc, …   5                                     [OK]
codenvy/ubuntu_jdk8                                       Ubuntu, JDK8, Maven
```

```
3, git, curl, nmap, mc, …   3                                          [OK]
darksheer/ubuntu                                          Base Ubuntu Image --
Updated hourly             3                                          [OK]
1and1internet/ubuntu-16-nginx-php-5.6-wordpress-4      ubuntu-16-nginx-
php-5.6-wordpress-4             2
[OK]
pivotaldata/ubuntu                                        A quick freshening-
up of the base Ubuntu doc…   1
ossobv/ubuntu                                             Custom ubuntu image
from scratch (based on o…   0
pivotaldata/ubuntu-gpdb-dev                               Ubuntu images for
GPDB development              0
1and1internet/ubuntu-16-healthcheck                       ubuntu-16-
healthcheck                            0
[OK]
smartentry/ubuntu                                         ubuntu with
smartentry                          0
[OK]
1and1internet/ubuntu-16-rspec                             ubuntu-16-rspec
0                                          [OK]
1and1internet/ubuntu-16-sshd                              ubuntu-16-sshd
0                                          [OK]
```

buradan NAME başlığı altındaki istediğimiz image i seçip docker pull <imageAdı> ile indirebiliriz.

Peki indiridiğimiz local deki Image ları nasıl görüntüleyebiliriz ? aşağıdaki komutlar ile bu işlemi gerçekleştirebiliriz ;

```
docker images
```

veya

```
docker image ls
```

Kendi bilgisayarımıza daha önce pull komutu ile indirdiğimiz image'ı nasıl silebiliriz.Wordpress image ını indirdiğimizi varsayalım. Kod Aşağıdaki gibi;

```
//docker'ın eski versiyonlarda ki yazım şekli
docker rmi wordpress
```

veya

```
//docker'ın yeni versiyonlardaki yazım şekli
docker image rmi wordpress
```

NOT: Burada Açıklamalarda Docker'ın eski ve yeni versiyonlarından bahsettim yeni versiyon Docker eski versiyon yazım şeklini de kabul etmektedir.Yeni versiyonlar geçmişe yönelik kodları destekler.Eski sürüm docker kullanıyorsanız doğal olarak yeni versiyon kod şekillerini uygulayamayacaksınız...

Lokale İndirilmiş olan veya indirilmemiş olan Imageları çalıştırabilmemizi sağlayan komut nedir ? Ubuntu image'ı üzerinden gidelim.

```
docker run ubuntu
```

Bu komut ile docker şunu yapıyor , Bu image daha önce indirilmiş mi buna bakıyor evet indirilmiş ise lokaldeki image dosyasını direk çalıştırıyor ve container ayağa kalkıp işlevi ne ise onu yapıyor.Şayet indirilmemiş ise ilk önce Docker registry den image ı indirip ardından container ı ayağa kaldırıyor.

Bu komuttan sonra çalışan Containerları nasıl görüntüleyebiliriz ? Daha önceki komutta ubuntu image ını çalıştırmıştık.Bakalım duruyormu ?

```
//docker'ın eski versiyonlarda ki yazım şekli
docker ps
```

veya

```
//docker'ın yeni versiyonlardaki yazım şekli
docker container ps
```

veya

```
//docker'ın yeni versiyonlardaki yazım şekli
docker container ls
```

Burada ki komutları yazdığınızda herhangi birşey'i ekranda göremeyeceksiniz "docker run <ImageAdı>" ile aslında çalış işlemini yap ve container'ı sonlandır dedik.
Peki çalışmış ve durmuş olan veya çalışan Container ları nasıl görüntüleyeceğiz ? Kod aşağıdaki gibi ;

```
//docker'ın eski versiyonlarda ki yazım şekli
docker ps -a
```

veya

```
//docker'ın yeni versiyonlardaki yazım şekli
docker container ps -a
```

veya

```
//docker'ın yeni versiyonlardaki yazım şekli
docker container ls -a
```

Yukarıda ki komutlardan birisi ile terminal ekranında aşağıda ki gibi bir çıktı verecek ;

```
CONTAINER ID        IMAGE               COMMAND             CREATED
STATUS                      PORTS               NAMES
951b5d527f75        ubuntu              "/bin/bash"         5 seconds ago
Exited (0) 3 seconds ago                        vigorous_albattani
```

Burada gördüğünüz üzere çalışmış işlevini yerine getirmiş ve durmuş bir container'ı görüntülüyoruz.Bu bizim biraz önce çalıştırdığımız ubuntu image'ı …Dikkat etmemiz gereken noktalardan birisi de her çalışan container a benzersiz bir id verilmesi(**CONTAINER ID**) bir diğeri ise çalıştırdığımız image a docker kendisi benzersiz bir isim veriyor biz belirtmediğimizden (**NAMES**) bende bu "vigorous_albattani" sizde farklı bir isim olabilir.**IMAGE** yazan noktada hangi **IMAGE** ı çalışmış ve durmuş olduğu bilgisi , **CREATED** da ne zaman işlemin yapıldığı bilgisi.STATUS kısmında ise container' ın durum bilgisi , **PORT** kısmında ise çalışmış olan container'ın dışarıya açık olan portlar bilgisi(Konuya daha sonra değineceğim) verilmektedir.STATUS kısmında container'ımızın sorunsuz olarak çalışıp çıkış yapıldığı bilgisi verilmektedir.

Daha önce çalıştırdığımız ve çıkış yapılan container ı nasıl silebiliriz ? Şimdi bu duruma göz atalım ;

```
//docker'ın eski versiyonlarda ki yazım şekli
docker rm &amp;lt;containerIDsi veya containerAdi&amp;gt;
```

veya

```
//docker'ın yeni versiyonlardaki yazım şekli
docker container rm &amp;lt;containerIDsi veya containerAdi&amp;gt;
```

NOT : Çalışan bir container ı silmek istediğimizde yukarıdaki komutu kullanır isek hata alacağız.Bu hata container in hala çalıştığını ve önce container ın durdurulması gerektiğini belirtecektir.Şayet çalışan container'ı zorla durdurmak istersek aşağıda ki komutları kullanabiliriz ;

```
//docker'ın eski versiyonlarda ki yazım şekli
docker rm -f &lt;containerIDsi veya containerAdi&gt;
```

veya

```
//docker'ın yeni versiyonlardaki yazım şekli
docker container rm -f &lt;containerIDsi veya containerAdi&gt;
```

veya

```
//docker'ın eski versiyonlarda ki yazım şekli
docker rm --force &lt;containerIDsi veya containerAdi&gt;
```

veya

```
//docker'ın yeni versiyonlardaki yazım şekli
docker container rm --force &lt;containerIDsi veya containerAdi&gt;
```

Çalışan containerların ID sini almak için aşağıdaki komutları kullanabilirsiniz ;

```
//docker'ın eski versiyonlarda ki yazım şekli
docker ps -q &lt;containerIDsi veya containerAdi&gt;
```

veya

```
//docker'ın yeni versiyonlardaki yazım şekli
docker container ps -q
```

veya

```
//docker'ın yeni versiyonlardaki yazım şekli
docker container ls -q
```

Çalışmış ve durmuş veya çalışan Containerların ID sini almak için aşağıdaki komutları kullanabilirsiniz ;

```
//docker'ın eski versiyonlarda ki yazım şekli
docker ps -aq &lt;containerIDsi veya containerAdi&gt;
```

veya

```
//docker'ın yeni versiyonlardaki yazım şekli
docker container ps -aq
```

veya

```
//docker'ın yeni versiyonlardaki yazım şekli
docker container ls -aq
```

Yukarıdaki komutlarda -aq parametrelerini açıklayacak olursak a-> Çalışan Çalışmayan Bütün Containerlar manasında q-> Sadece ID leri göster manasındadir.Komut yazımı docker esnektir isterseniz “-aq” isterseniz “-a -q” şeklinde parametreleri tanımlayabilirsiniz … bu esnekliği daha da abartmak istersek docker deamon dan bunu öğrenebiliriz Nasıl mı ?

```
docker container ls --help
```

```
```

veya

```
docker container ps --help
```

veya

```
docker ps --help
```

NOT: Bundan sonraki örneklerde yeni versiyon kod yazımı ile devam edeceğim.İsterseniz eski tip yazım tarzı ile de yukarıdaki örneklere benzer şekilde kodları yazabilirsiniz …

Çalışan bir container ı nasıl durdurabiliriz.Durdurabilirizden kastım **PAUSE** işlemi ;

```
docker container pause &lt;ContainerID veya ContainerAdi&gt;
```

Pause edilen bir Container ı kaldığı yerden devam ettirmek için ;

```
docker container unpause &lt;ContainerID veya ContainerAdi&gt;
```

Çalışan container ı tamamen durdurmak için iki yol mevcut stop veya kill komutları ile … burada dikkat etmemiz gereken husus stop komutu container ın düzgün şekilde çalışmayı durdurması için kullanılırken kill komutu ile container ı düzgün durdurulması için zaman vermeden container ı durduracaktır.Uygulanışları aşağıdaki gibidir ;

STOP komutu ile container durdurma

```
docker container stop &lt;ContainerID veya ContainerAdi&gt;
```

KILL komutu ile container durdurma

```
docker container kill &lt;ContainerID veya ContainerAdi&gt;
```

Çalışan container ı silmek için yukarıdaki komuttaki gibi önce durdurup sonra silmek önerilir.-f veya –force komutu ile durdurmadan da silebiliriz.Uygulanışları aşağıdaki gibidir ;

Çalışan container ı önerilen şekilde silme ;

```
docker container stop &lt;ContainerID veya ContainerAdi&gt;
docker container rm &lt;ContainerID veya ContainerAdi&gt;
```

Çalışan container ı zorla silme

```
docker container rm -f &lt;ContainerID veya ContainerAdi&gt;
```

Çalışmakta olan container içerisinde şu anda çalışmakta olan işlemleri görmek için linuxtan bildiğiniz TOP komutunu kullanıyoruz.Uygulanışı ;

```
docker top &lt;ContainerID veya ContainerAdi&gt;
```

Bir Container'ı sürekli arka planda çalışmasını istediğimizde uygulanan komut ;

```
docker run -it -d &lt;ImageID veya ImageAdi&gt;
```

Sürekli çalışırken diye bahsettim ama nasıl sürekli container ın arka planda çalıştırılabileceğini anlatmayı atlamışım üstte anlatıyor

Container çalıştırma ile ilgili daha fazla seçenek var elbette … Aslına bakarsanız Docker bunların hepsini bize açıklıyor soralım beraber ;

```
docker run --help
```

Gelen çıktıda tüm container çalıştırma ile ilgili kullanım yöntemlerini açıklamaları ile birlikte görebilirsiniz …

Bir container veya image ile ilgili bilgi almak istersek aşağıdaki gibi komut girebiliriz ;

```
docker container inspect &lt;ContainerID veya ContainerAdi veya imageID
veya imageAdı&gt;
```

Tüm Çalışan Çalışmayan Containerları topluca silmek için aşağıdaki komutu kullanabilirsiniz ;

```
docker rm -f $(docker container ls -a)
```

Bütün Localde ki Imageları silmek için aşağıdaki komutu kullanabilirsiniz ;

```
docker rmi -f $(docker container ls -a)
```

Çalışan bütün containerları durdurmak için ;

```
docker container stop $(docker container ls -a)
```

Çalışan bir container içerisinde komut koşturmak için aşağıdaki kodu uygulayabiliriz ;

```
docker exec &lt;ContainerID veya ContainerAdi&gt;  &lt;Komut&gt;
// örn : docker exec ubuntu ls
```

Çalışan bir Container içinde terminal açmak için örnek kod. (İlgili Image'da /bin/bash bulunduğu varsayımı ile).

```
docker exec -it &lt;ContainerID veya ContainerAdi&gt;  /bin/bash
// örn : docker exec -it myubuntu /bin/bash
```

Container ile ilgili logları görüntülemek istersek uygulanışı aşağıdaki gibidir ;

```
docker logs &lt;ContainerID veya ContainerAdi&gt;
```

Containerın çalışma anında yazılan logları da anlık görüntülemek istersek "-f" yani follow parametresini kullanıyoruz , Uygulanışı aşağıdaki gibidir ;

```
docker logs -f &lt;ContainerID veya ContainerAdi&gt;
```

Öğrenmenin yolu uygulamaktan geçer malum.Sadece okumak ile olmuyor Uyguladığınızı varsayıyorum Aklıma gelen temel komutları açıklamaya çalıştım eksik olduğunu fark ettiğiniz temel komut var ise bilgi verin ekleyeyim

Bir sonraki yazım ;

Docker Image Oluşturma İşlemi(Tıklayınız)

Umarım faydalı olmuştur ?

Serkan Kaya
Full Stack Java Developer