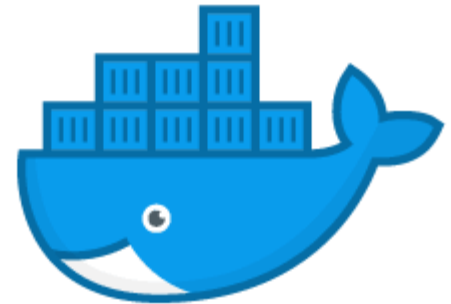


Docker

Riccardo Cattaneo

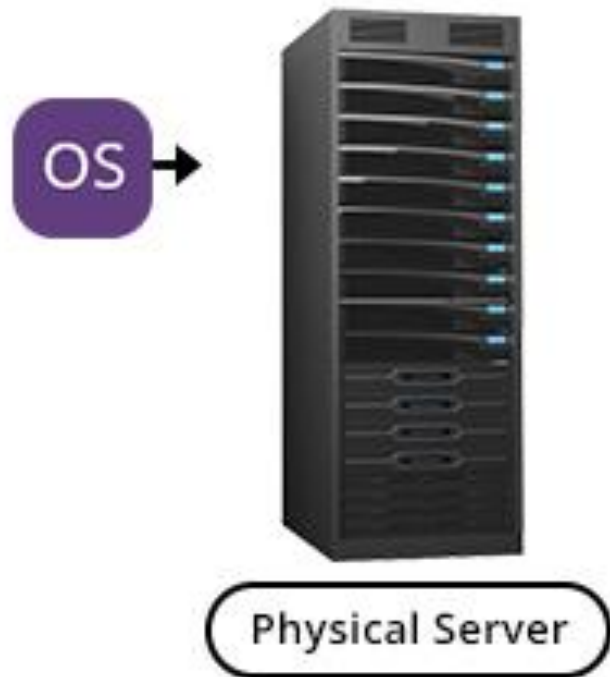


docker

Introduzione

Inizialmente ogni applicazione veniva eseguita su un server fisico, che stava a significare un enorme spreco di risorse. Successivamente ci siamo orientati verso la **virtualizzazione** che ha reso le cose sicuramente migliori... ma c'è un problema...

Il problema è che ogni macchina virtuale necessita del proprio sistema operativo e come ben sappiamo ogni sistema operativo consuma risorse in termini di CPU, RAM e spazio su disco.



VS



Introduciamo quindi il concetto di **container**. I container ci offrono la possibilità di “pacchettizzare” un’applicazione e renderla trasportabile e quindi disponibile verso altri sistemi.

Una delle principali differenze rispetto alla virtualizzazione è che i container all’interno dello stesso host condividono lo stesso Sistema operativo.

Container

Un container è un ambiente isolato ed è proprio l'isolamento e la sicurezza che ne derivano che ci permettono di eseguire più container simultaneamente all'interno di un host.

I container sono “leggeri” in quanto non necessitano della presenza di un Hypervisor come nel caso delle macchine virtuali. Si eseguono direttamente all'interno del kernel della macchina host.

Possiamo immaginare i container come delle piccole entità al cui interno viene eseguito un'applicazione in maniera autonoma: ovvero quell'applicazione che contiene tutti i componenti essenziali affinché possa eseguirsi correttamente.

Ciò rende i container trasportabili che è uno dei maggiori punti di forza di questa tecnologia.

Docker è un progetto OpenSource e quindi il codice è pubblicamente disponibile. Si può trovare su GitHub sotto il nome di «Moby Project».

La maggior parte del progetto è scritto nel linguaggio di programmazione Golang (GO) che è il linguaggio sviluppato da Google.

Si può operare con i container da differenti punti di vista, da una parte abbiamo la **gestione sistemistica** e dall'altra lo **sviluppo del software**.

Se facciamo riferimento all'ambito sistemistico ci occuperemo soprattutto della gestione dei container (creazione, start, stop, eliminazione, comunicazione di rete ecc.).

Se facciamo riferimento all'ambito dello sviluppo ci soffermeremo maggiormente sull'applicazione, su come trasformarla in un container e delle funzionalità che dovrà avere.

Tipologie di Container

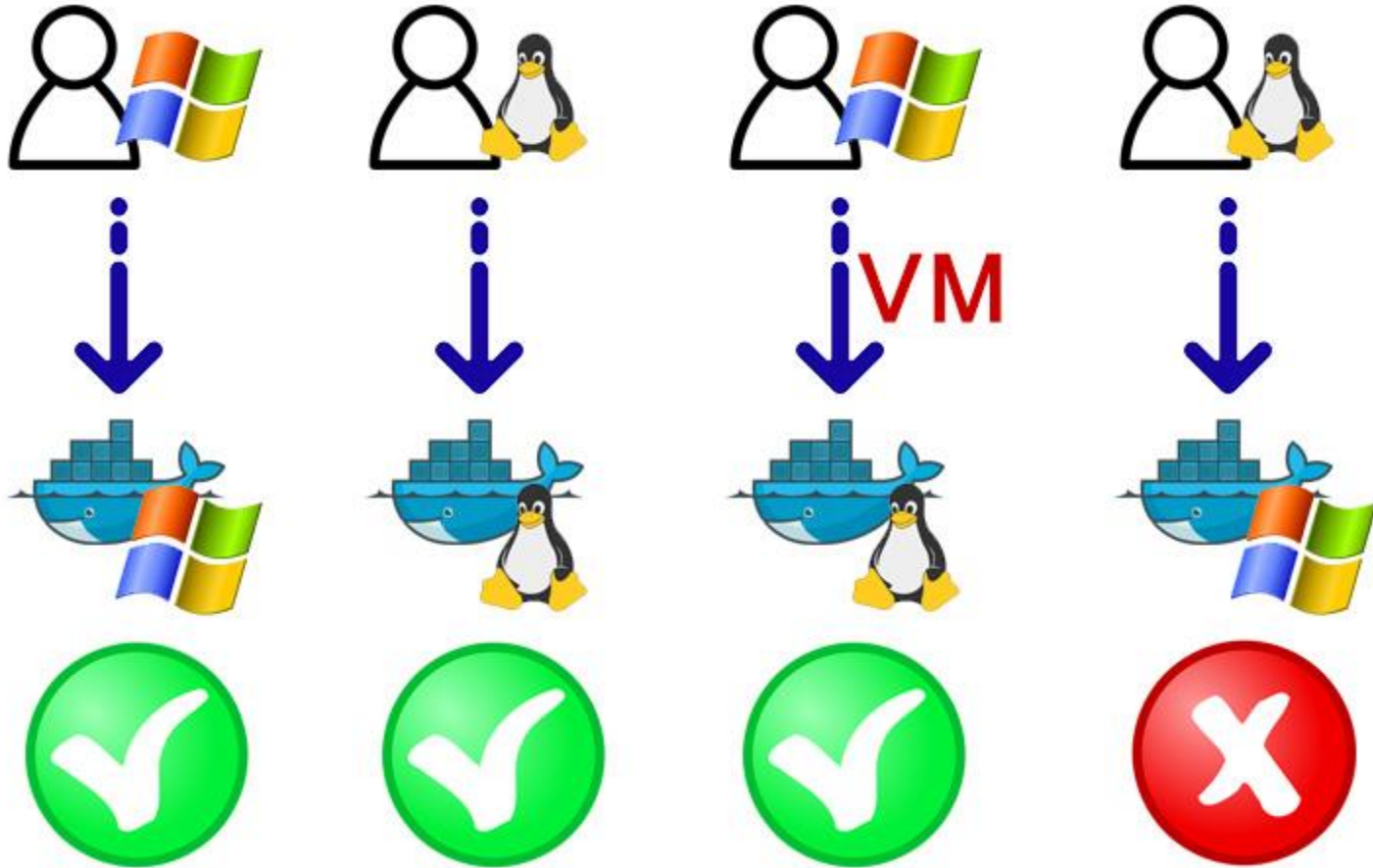
Attualmente abbiamo due tipologie di container docker :

- **Container Linux** che si eseguono su sistemi linux;
- **Container Windows** che si eseguono su sistemi Windows, nello specifico su Windows 10 oppure Windows Server 2016.

Queste tipologie sono differenti tra loro. Un container Linux si può eseguire solo su macchina Linux e un Sistema Windows si può eseguire solo su macchina Windows.

E' possibile eseguire Container Linux su Windows o Mac installando Docker, che a sua volta, installa una **macchina virtuale Linux** così da poter emulare il kernel Linux ed eseguire i container.

E' possibile installare docker su un server fisico, su una macchina virtuale oppure sfruttando gli ambienti cloud. Non ci sono limiti.



Installazione su sistemi Windows

L'installazione su windows si può differenziare in due tipologie : l'installazione che ci permetterà di eseguire dei container nativi windows e l'installazione che ci permetterà di eseguire container linux.



La versione di docker per windows è “Docker for Windows” ed è la versione che installa la macchina virtuale ed esegue container linux ed è disponibile solo su **Windows 10 64bit**, Windows Server 2016 e version successive.

Deve essere abilitata la funzionalità **Hyper-V** in quanto richiesta da Docker per eseguire la macchina virtuale che ci permetterà di eseguire i container Linux su Windows. L'installer lo farà in automatico.

Ed infine deve essere abilitata la virtualizzazione da BIOS. Solitamente è abilitata di default.

 Installing Docker Desktop 4.25.0 (126437)



Configuration

- ☒ Use WSL 2 instead of Hyper-V (recommended)
- ☒ Add shortcut to desktop

Ok

WSL 2

Da dicembre 2020 Docker Desktop si è legato con WSL 2, la seconda versione di Windows Subsystem for Linux ovvero il componente software basato su Microsoft Hyper-V per Windows 10 e Windows 11 che consente l'esecuzione delle principali distribuzioni Linux anche in modalità grafica.

Per verificare la corretta installazione lanciare il seguente comando :

- `C:\> docker --version`

Docker Desktop

Upgrade plan

Search for images, containers, volumes, extensions and more...

Ctrl+K

#

⚙

Sign in

—

□

✕

Containers

Images

Volumes

Dev Environments BETA

Docker Scout

Learning center

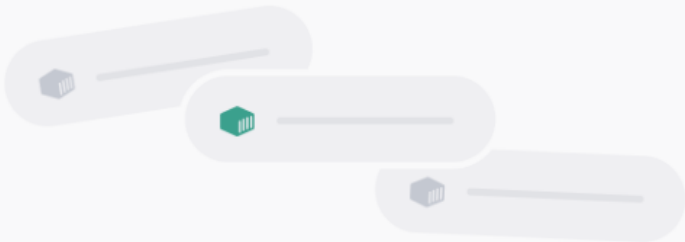
Extensions

⋮

⊕ Add Extensions


Containers

[Give feedback](#)



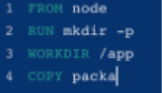
Your running containers show up here

A container is an isolated environment for your code




What is a container?

5 mins




How do I run a container?

6 mins



Run Docker Hub images

5 mins



Multi-container applications

6 mins

[View more in the Learning center](#)

Engine running

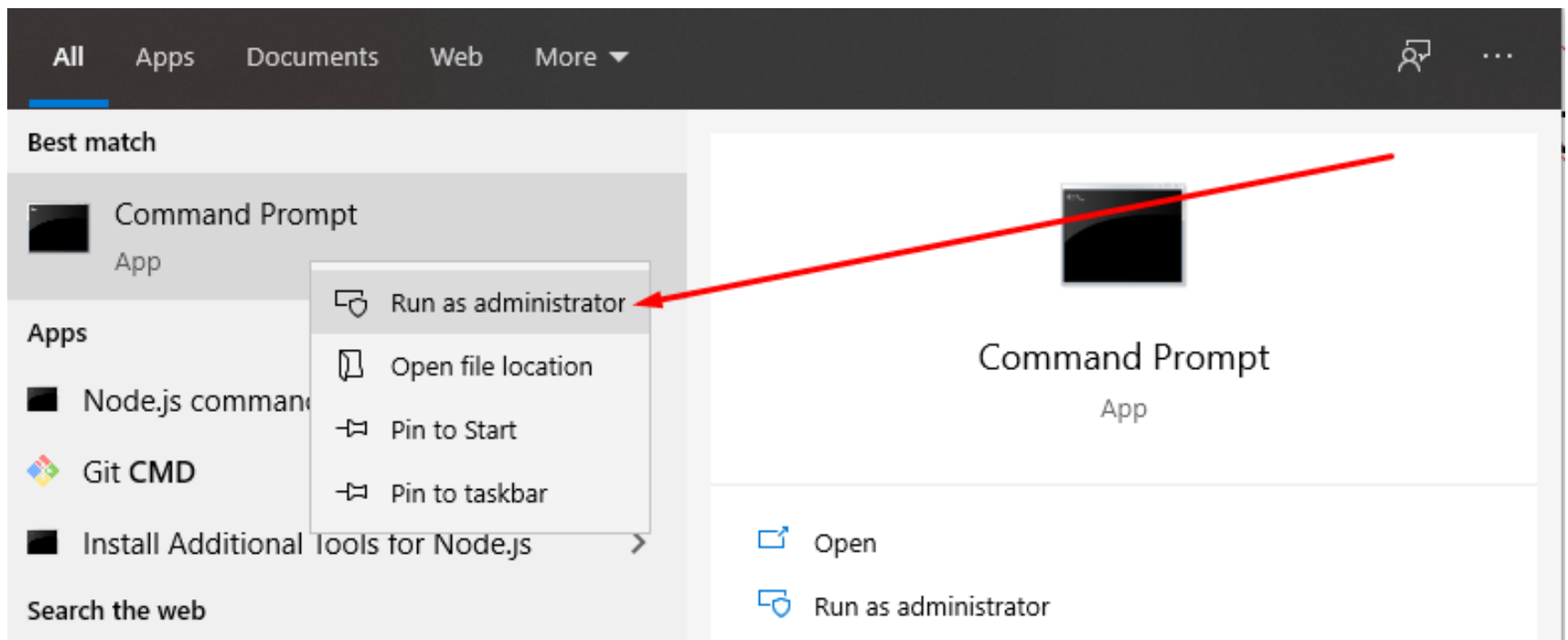
▶ || 🔌 ⚙

RAM 2.43 GB CPU 0.00% 🔌 Not signed in

v4.25.0

Docker

Per verificare che l'installazione è andata a buon fine apriamo il prompt dei comandi di windows in modalità amministrazione :



Microsoft Windows [Version 10.0.18363.1082]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>docker version

Client: Docker Engine - Community

Cloud integration 0.1.18

Version: 19.03.13

API version: 1.40

Go version: go1.13.15

Git commit: 4484c46d9d

Built: Wed Sep 16 17:00:27 2020

OS/Arch: windows/amd64

Experimental: false

Server: Docker Engine - Community

Engine:

Version: 19.03.13

API version: 1.40 (minimum version 1.12)

Go version: go1.13.15

Git commit: 4484c46d9d

Built: Wed Sep 16 17:07:04 2020

OS/Arch: linux/amd64

Experimental: false

containerd:

Version: v1.3.7

GitCommit: 8fba4e9a7d01810a393d5d25a3621dc101981175

runc:

Version: 1.0.0-rc10

GitCommit: dc9208a3303feef5b3839f4323d9beb36df0a9dd

docker-init:

Version: 0.18.0

GitCommit: fec3683

Switch Linux / Windows

I comandi appena digitati sono stati eseguiti sul server Linux, mentre per eseguire container nativi windows bisogna cambiare modalità : tasto destro su icona docker, e poi “switch to windows container” e verificare con docker version.



Installazione su sistemi Linux

```
sudo apt update
```

```
sudo apt install apt-transport-https ca-certificates curl software-properties-common
```

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

```
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable"
```

```
sudo apt update
```

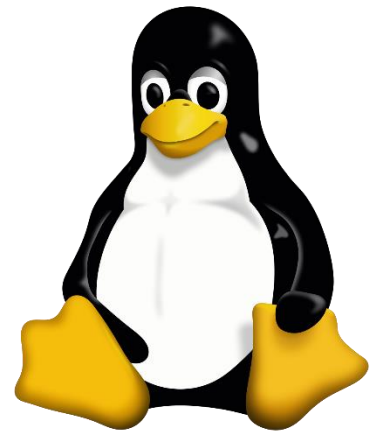
```
apt-cache policy docker-ce
```

```
sudo apt install docker-ce
```

```
sudo systemctl status docker
```

```
sudo docker version
```

```
sudo docker run hello-world
```



Dopo aver installato Docker si suggerisce di compiere due ulteriori passi :

- Gestione di Docker come utente non root
- Configurare Docker per avviarsi all'avvio

Gestire docker come utente-non-root

- 1) creazione gruppo docker -> `sudo groupadd docker`
- 2) aggiunta utente al gruppo Docker -> `sudo usermod -aG docker $USER`
- 3) riavvio host.
- 4) test -> `docker run hello-world`

Abilitare docker per avviarsi all'avvio

- 1) `sudo systemctl enable docker`