An alternative for Docker Desktop

Setup Docker on WSL 2 guideline

Version 1.0
Aug 2022

Author: Technical team
Security Classification:

3 Noble Street, London, United Kingdom, EC2V 7DE
Tel: +44 (0)20 7333 0033
Email: info@nashtechglobal.com

Nash Tech.

Table of Contents

Note:

- ***Not support the Windows containers***
- The guideline only works on WSL2
- To enable and run WSL 2,
    - you must be running Win 10 version 2004 and higher (Build 19041 and higher) or Win 11
    - Enable virtualization in Bios

Acronym

WSL 2	Windows Subsystem for Linux, version 2
Kind	Kubernetes IN Docker
VS code	Visual Code

1. Introduction

Docker is an open-source project that automates the development, deployment and running of applications inside isolated containers. Containers allow developers to bundle up an application with all of the parts it needs, such as libraries and other dependencies, and ship it as one package

> On Linux, you can install the Docker engine that is available on a variety of Linux distro such as Ubuntu, Debian… ref to Install Docker Engine on Ubuntu | Docker Documentation **(*)**
>
> On Windows, using Docker desktop on Windows with WSL 2 backend option, ref to Install Docker Desktop on Windows | Docker Documentation

However, Docker updated its Docker desktop license agreement (if your company has 250+ employees or more than $10 million in annual revenue you will not able to use Docker Desktop without a paid subscription. It remains free for smaller companies, private use, open-source projects, and educational purposes). This license update is only related to Docker Desktop and not to Docker or the Docker Engine**. *This means that NashTech cannot use Docker desktop in developing environment freely.***

The solution ☞ using Docker on WSL 2 without Docker desktop

Docker on WSL2 (*Windows Subsystem for Linux, version 2*): it is a mixed option, enables Docker engine running in Windows system without Docker desktop license. However, it will require an additional procedure – install Linux image named WSL 2 inside this Windows system. It acts as using Docker engine on Linux.

Below sections will guide us how to setup it step by step.

2. Setup WSL2

This section can be skipped if you installed WSL2 on your PC

2.1 Enable Windows feature

2.1.1 Enable the Windows Subsystem for Linux

Open PowerShell as **Administrator** (**Start menu > PowerShell > right-click > Run as Administrator**) and enter this command:

```
dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart
```

2.1.2 Enable Virtual Machine feature

Before installing WSL 2, you must enable the Virtual Machine Platform optional feature. Your machine will require virtualization capabilities to use this feature.

Open PowerShell as Administrator and run:

```
dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart
```

2.2 Install WSL 2

Make sure WSL is version 2

```
Wsl –set-default-version 2
```

Install the Ubuntu distribution of Linux on WSL 2

```
Wsl –install -d ubuntu
```

To list of installed Linux distribution on the WSL environment

```
Wsl -l -v
```

In order to setup manually, ref to Manual installation steps for older versions of WSL | Microsoft Docs

2.3 Upgrade packages

```
sudo apt update && sudo apt upgrade
```

If the upgrade command succeeded, please skip the below configuration.

If the command fail to access package servers, change DNS resolver

```
echo -e "[network]\ngenerateResolvConf = false" | sudo tee -a /etc/wsl.conf

sudo unlink /etc/resolv.conf

echo nameserver 1.1.1.1 | sudo tee /etc/resolv.conf
```

3. Setup Docker

3.1 Package repository configuration

```
. /etc/os-release

curl -fsSL https://download.docker.com/linux/${ID}/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.asc
```

ID will be “ubuntu”, depending on what is in /etc/os-release

Add and update the repo information so that apt will use it in the future:

```
echo "deb [arch=amd64] https://download.docker.com/linux/${ID} ${VERSION_CODENAME} stable" | sudo tee
/etc/apt/sources.list.d/docker.list

sudo apt update
```

3.2 Install Docker

```
sudo apt install docker-ce docker-ce-cli containerd.io
```

3.3 Add user to docker group

The service (dockerd) and client (docker) communicate over a socket and/or a network port. For communication over the socket, privileged access is required. Two ways to obtain this access

- Run docker as root (sudo docker)
- Create a Unix group called ***docker*** and add users to it. When the Docker daemon starts, it creates a Unix socket accessible by members of the docker group.

```
sudo usermod -aG docker $USER
```

3.4 Launch *dockerd*

Pick the name of the distro such as Ubuntu

```
Wsl -l -q
```

Adding following lines in ***.bashrc*** or ***.profile*** or *.zshrc* to lauch dockerd automatically

Such as ***vi ~/.bashrc***

```
DOCKER_DISTRO="Ubuntu"
DOCKER_LOG_DIR=$HOME/docker_logs
mkdir -pm o=,ug=rwx "$DOCKER_LOG_DIR"
/mnt/c/Windows/System32/wsl.exe -d $DOCKER_DISTRO sh -c "nohup sudo -b dockerd < /dev/null >
$DOCKER_LOG_DIR/dockerd.log 2>&1"
```

3.5 Passwordless launch of *dockerd*

If the above script is placed in .bashrc (most Linux distros) or .profile ..., you will likely be prompted for a password most every time a new terminal window is launched.

To skip this, enter ***sudo visudo*** and add the below line

```
%docker ALL=(ALL)  NOPASSWD: /usr/bin/dockerd
```

3.6 Run Docker from Windows

Configuring at **/etc/docker/daemon.json**; create ***docker*** folder if it is not exist

```
{
        "hosts": ["tcp://127.0.0.1:2375", "unix:///var/run/docker.sock"]
}
```

From windows terminal, run the following command to create the docker context to docker in WSL 2

```
docker context create wsl-contect --docker "host=tcp://127.0.0.1:2375"
```

Verify

```
docker info
```

Note: to download docker CLI on Windows, ref to StefanScherer/docker-cli-builder: Build Docker CLI for Windows (github.com) - ***optional***

4. Development environment preparation

In case you need to leverage docker but without using docker desktop, you can consider 2 options to prepare your working environment

Option 1: all source code will be stored on Windows and connect to external docker applications such as Postgres, RabbitMQ…

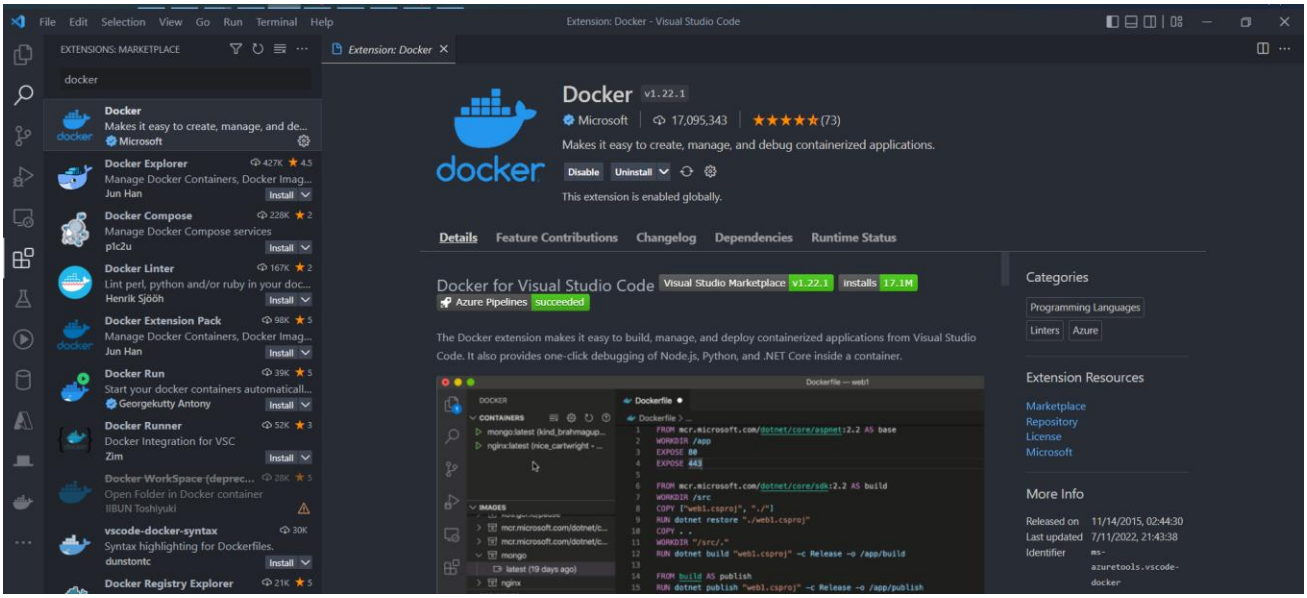
- Pros:
  - Source code is managed on Windows
  - Similar the working environment with docker desktop
- Cons:
  - The limitation about mount volume

Option 2: develop inside WSL 2

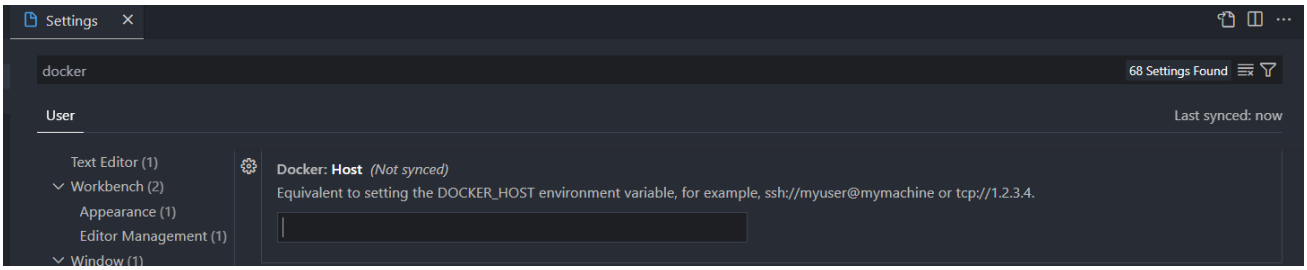
- Pros:
  - Fully development in Linux environment
  - Natively docker development
- Cons:
  - Must know the basic Linux commands
  - Source code is stored inside WSL2

5. Develop with Visual Code

Install the Docker extension



Configure the setting of the Docker extension



6. Kubernetes – K8S

To use K8S in local development without Docker desktop, Kind is a choice. kind is a tool for running local Kubernetes clusters using Docker container “nodes”.

kind was primarily designed for testing Kubernetes itself, but may be used for local development or CI.

6.1 Install Kind on WSL 2

Ref to kind – Quick Start (k8s.io)

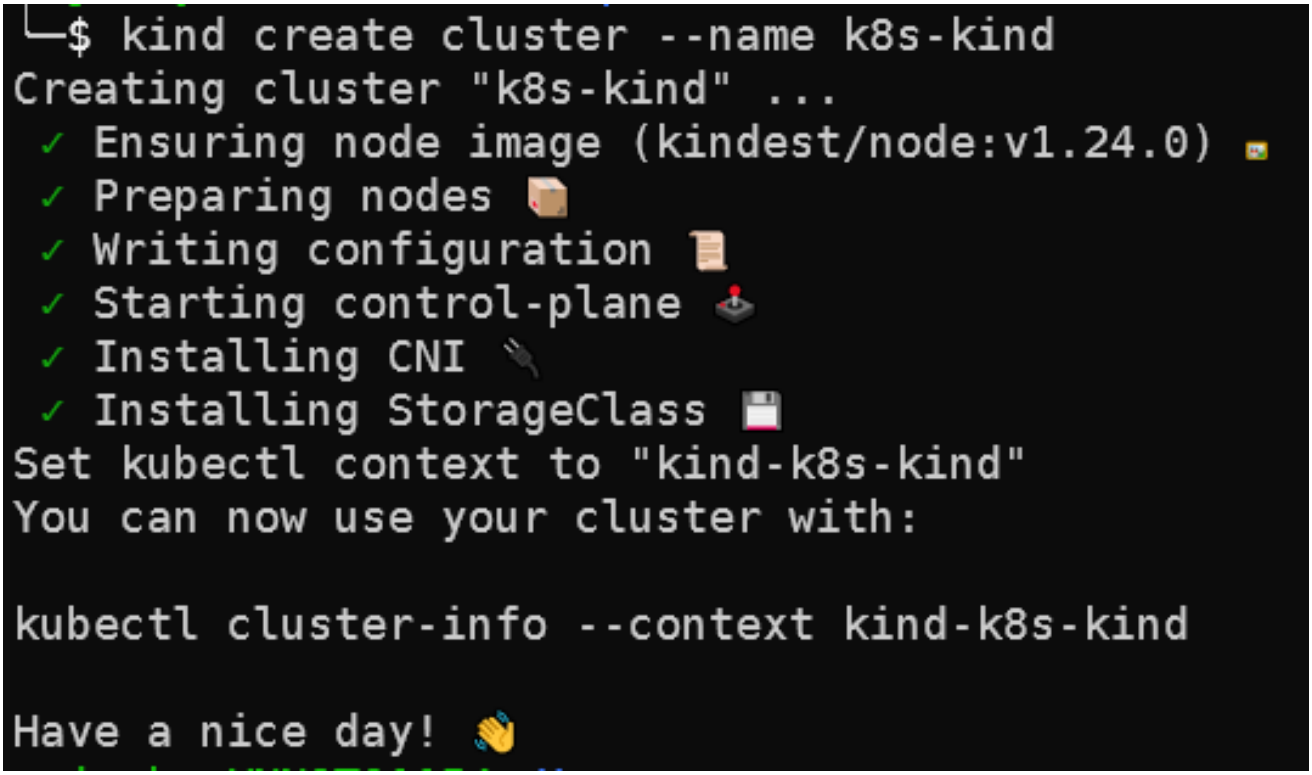
```
curl -Lo ./kind https://kind.sigs.k8s.io/dl/v0.14.0/kind-linux-amd64
chmod +x ./kind
sudo mv ./kind /usr/local/bin/kind
```

Verify

```
kind version
```

6.2 Creating a cluster

```
kind create cluster --name k8s-kind
```



6.3 Interacting With Your Cluster

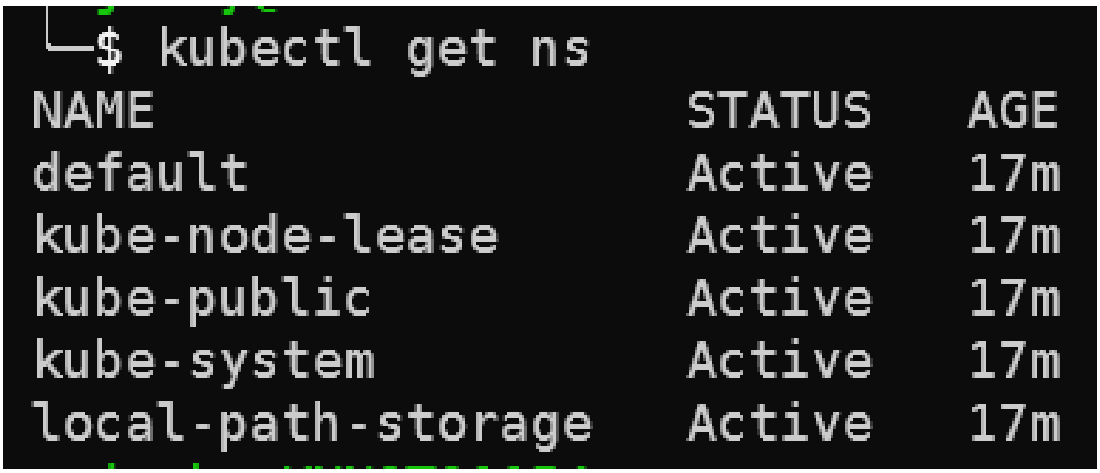
```
kind get clusters
```

Get nodes in the Kind cluster

```
kubectl get nodes
```

Get all namespace

```
kubectl get ns
```



Get all services

```
kubectl get svc --all-namespaces
```

```
└─$ kubectl get svc --all-namespaces
NAMESPACE     NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)                  AGE
default       kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP                  19m
kube-system   kube-dns     ClusterIP   10.96.0.10   <none>        53/UDP,53/TCP,9153/TCP   18m
```

7. Reference

WSL 2 Windows Subsystem for Linux Documentation | Microsoft Docs

Docker engine Install Docker Engine | Docker Documentation

Kind kind (k8s.io)