

# Docker (Engine/CLI) WSL Installation Guide

## Why WSL Docker?

Docker changed their [Subscription Service Agreement](#) and Docker Desktop requires paid subscription after Jan, 2022. But Docker Engine and CLI as open source containerization technologies still remain free.

To enable local docker-based development and testing, please follow the guide here to setup WSL-based docker to your work machine.

## What is WSL?

Windows Subsystem for Linux (WSL) is nothing but is a compatibility layer for running Linux (binary executables) natively on Suncor standard development machine (Win10 Surface).

To learn more, please refer to our [DPC WSL page for details](#).

## How WSL Docker?

Estimated time for installation: < 30 minutes

### Prerequisites

- Unmanaged Access

Please submit request through Service Portal: General Access Administration Request following guide Provisioning: Local Admin Rights and Unmanaged Access Process if you do not have.

- WSL 2

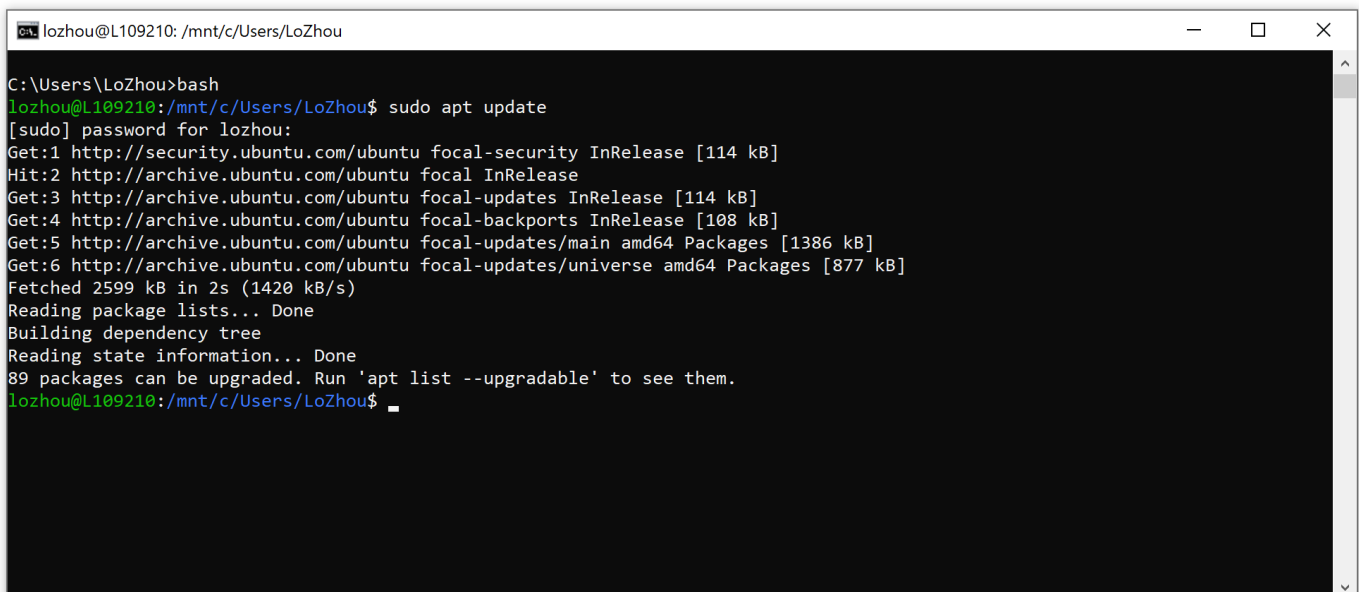
Please refer to [Windows Subsystem for Linux \(WSL\) Installation Guide](#) to install WSL 2. Please ensure you have WSL 2 instead of WSL 1.

### Installation

Installation of Docker to WSL is quick and simple.

#### 1. Update

Open PowerShell or Command Line Prompt as Administrator, type `$bash` to enter to WSL Linux mode. And then update local packages with command `$sudo apt update`, as follows:



```
lozhou@L109210: /mnt/c/Users/LoZhou
C:\Users\LoZhou>bash
lozhou@L109210:/mnt/c/Users/LoZhou$ sudo apt update
[sudo] password for lozhou:
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Hit:2 http://archive.ubuntu.com/ubuntu focal InRelease
Get:3 http://archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:4 http://archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:5 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [1386 kB]
Get:6 http://archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [877 kB]
Fetched 2599 kB in 2s (1420 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
89 packages can be upgraded. Run 'apt list --upgradable' to see them.
lozhou@L109210:/mnt/c/Users/LoZhou$
```

#### 2. Install Docker

Type in command `$sudo apt install docker.io -y` to install docker.

```
lozhou@L109210: /mnt/c/Users/LoZhou$ sudo apt install docker.io -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
docker.io is already the newest version (20.10.7-0ubuntu5~20.04.2).
0 upgraded, 0 newly installed, 0 to remove and 89 not upgraded.
lozhou@L109210: /mnt/c/Users/LoZhou$
```

This installation will also download and setup docker daemon program into directory `/usr/bin/dockerd`

### 3. Add Username to Docker Group

Type command into terminal `$sudo usermod -a -G docker USER` replacing USER with your username, for example:

```
lozhou@L109210: /mnt/c/Users/LoZhou$ sudo usermod -a -G docker lozhou
lozhou@L109210: /mnt/c/Users/LoZhou$
```

Now the work is all done.

Check your docker installation by checking its version `$docker -version`

```
lozhou@L109210: /mnt/c/Users/LoZhou$ docker -version
unknown shorthand flag: 'e' in -ersion
See 'docker --help'.

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Options:
  --config string      Location of client config files (default "/home/lozhou/.docker")
  -c, --context string  Name of the context to use to connect to the daemon (overrides DOCKER_HOST env var and default context set with "docker context use")
  -D, --debug           Enable debug mode
  -H, --host list       Daemon socket(s) to connect to
  -l, --log-level string Set the logging level ("debug"|"info"|"warn"|"error"|"fatal") (default "info")
  --tls               Use TLS; implied by --tlsverify
  --tlscacert string   Trust certs signed only by this CA (default "/home/lozhou/.docker/ca.pem")
  --tlscert string     Path to TLS certificate file (default "/home/lozhou/.docker/cert.pem")
  --tlskey string       Path to TLS key file (default "/home/lozhou/.docker/key.pem")
  --tlsverify          Use TLS and verify the remote
  -v, --version         Print version information and quit

Management Commands:
  builder      Manage builds
  config       Manage Docker configs
  container    Manage containers
```

\*Docker Daemon Auto Start

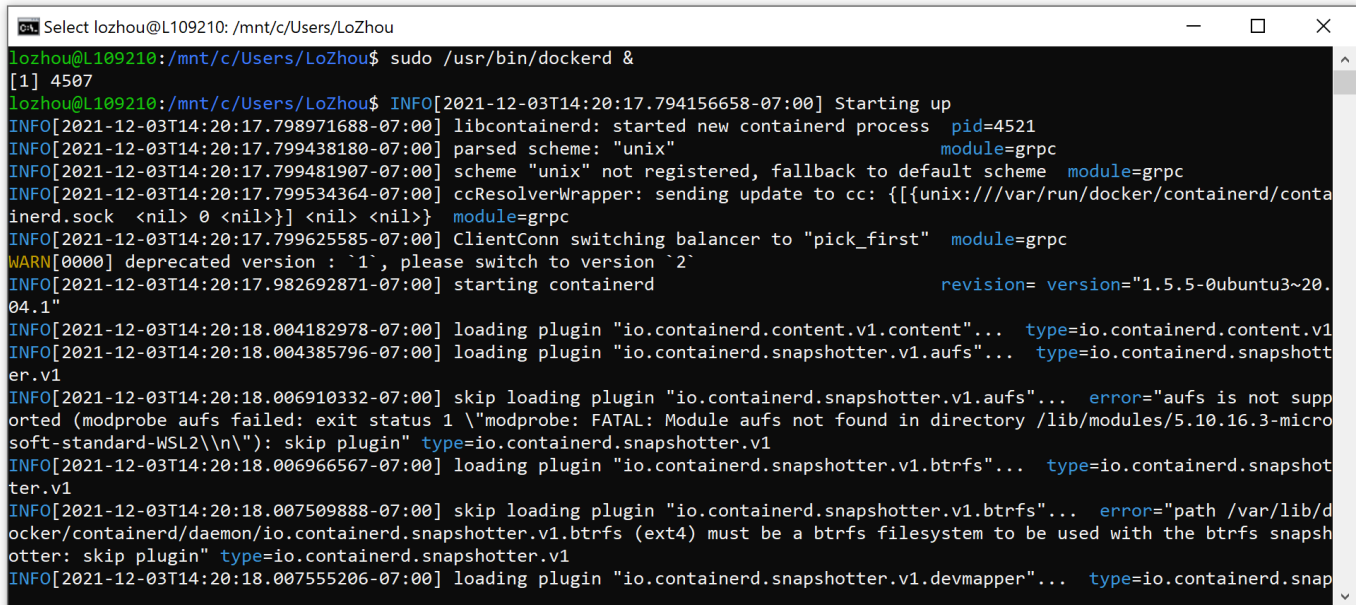
Please refer to page: [How to automatically start the Docker daemon on WSL2](#)

This step is nice to have, but not mandatory to use docker in WSL. If you do not want to run Docker Daemon in the background every time when you use WSL, just skip this step.

## How to Use Docker in WSL?

In Linux mode, Start and run docker daemon in the background. `&` is used to run it in the background, then you do not have to have a terminal open all the time for daemon. But if you want to the docker daemon logs, just remove `&`.

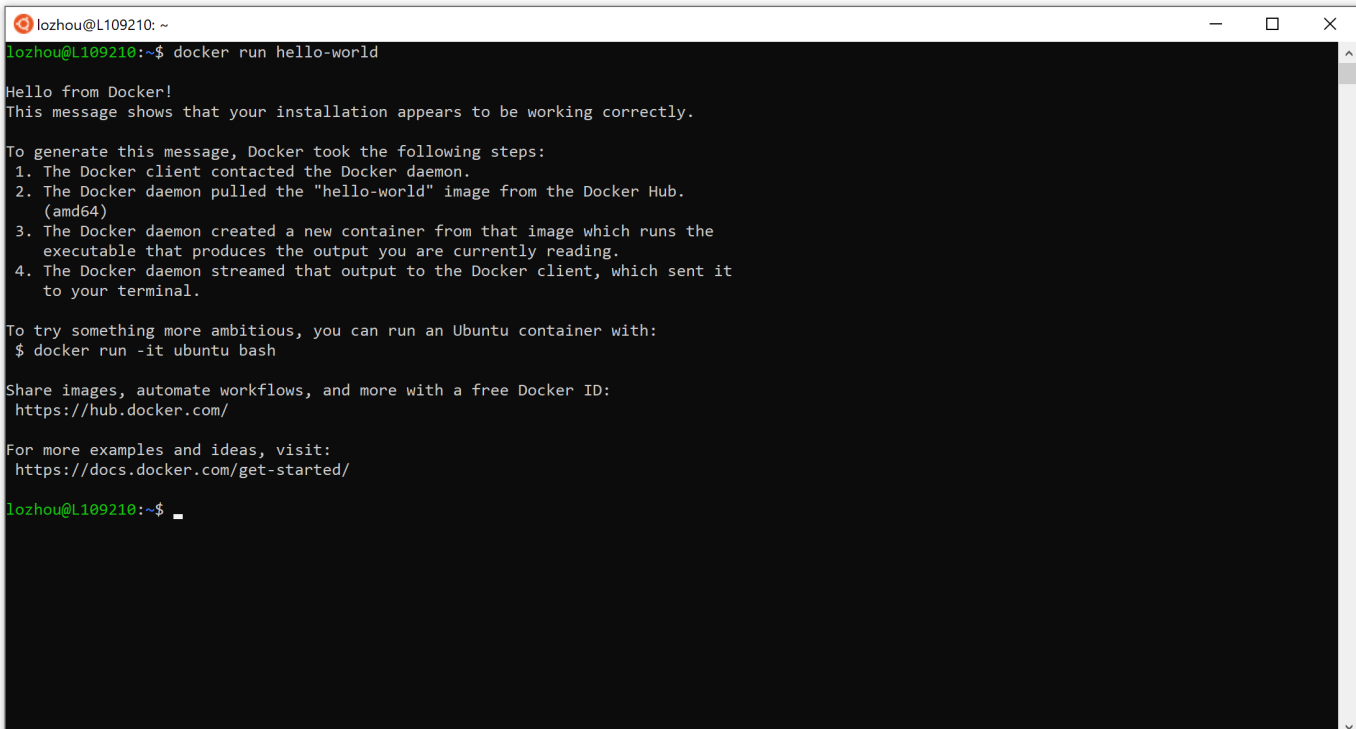
```
sudo /usr/bin/dockerd &
```



```
lozhou@L109210: /mnt/c/Users/LoZhou$ sudo /usr/bin/dockerd &
[1] 4507
lozhou@L109210: /mnt/c/Users/LoZhou$ INFO[2021-12-03T14:20:17.794156658-07:00] Starting up
INFO[2021-12-03T14:20:17.798971688-07:00] libcontainerd: started new containerd process pid=4521
INFO[2021-12-03T14:20:17.799438180-07:00] parsed scheme: "unix" module=grpc
INFO[2021-12-03T14:20:17.799481907-07:00] scheme "unix" not registered, fallback to default scheme module=grpc
INFO[2021-12-03T14:20:17.799534364-07:00] ccResolverWrapper: sending update to cc: {[{unix:///var/run/docker/containerd/conta
inerd.sock <nil> 0 <nil>}] <nil> <nil>}] module=grpc
INFO[2021-12-03T14:20:17.799625585-07:00] ClientConn switching balancer to "pick_first" module=grpc
WARN[0000] deprecated version : `1`, please switch to version `2`
INFO[2021-12-03T14:20:17.982692871-07:00] starting containerd revision= version="1.5.5-0ubuntu3~20.
04.1"
INFO[2021-12-03T14:20:18.004182978-07:00] loading plugin "io.containerd.content.v1.content"... type=io.containerd.content.v1
INFO[2021-12-03T14:20:18.004385796-07:00] loading plugin "io.containerd.snapshotter.v1.aufs"... type=io.containerd.snapshott
er.v1
INFO[2021-12-03T14:20:18.006910332-07:00] skip loading plugin "io.containerd.snapshotter.v1.aufs"... error="aufs is not supp
orted (modprobe aufs failed: exit status 1 \"modprobe: FATAL: Module aufs not found in directory /lib/modules/5.10.16.3-micro
soft-standard-WSL2\\n\\n\")": skip plugin" type=io.containerd.snapshotter.v1
INFO[2021-12-03T14:20:18.006966567-07:00] loading plugin "io.containerd.snapshotter.v1.btrfs"... type=io.containerd.snapshott
er.v1
INFO[2021-12-03T14:20:18.007509888-07:00] skip loading plugin "io.containerd.snapshotter.v1.btrfs"... error="path /var/lib/d
ocker/containerd/daemon/io.containerd.snapshotter.v1.btrfs (ext4) must be a btrfs filesystem to be used with the btrfs snapsh
otter: skip plugin" type=io.containerd.snapshotter.v1
INFO[2021-12-03T14:20:18.007555206-07:00] loading plugin "io.containerd.snapshotter.v1.devmapper"... type=io.containerd.snap
```

Then using docker in Windows WSL is just liking using docker CLI in any Linux OS. You could see complete docker commands here: [Use the Docker command line](#)

For example, run [hello-world](#) with `$docker run hello-world`



```
lozhou@L109210: ~$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

lozhou@L109210: ~$
```