

SSAFY Docker 설치 가이드 문서

부울경 8기 최홍준

목차

- 1. 개요
 - 1.1. 도커 엔진과 도커 데스크탑의 차이
 - 2. 시작하기
 - 2.1. 리눅스 환경에서 도커 엔진 설치하기
 - 2.1.1. AWS EC2 환경에서 도커 설치하기
 - 2.1.2. VMware의 리눅스 가상환경에서 설치하기
 - 2.2. 윈도우 환경에서 WSL2을 활용하여 설치하기
 - 3. 주요 명령어
-

1. 개요

도커란 가상머신과 유사하게 동작하는 듯 하지만 각 게스트 머신마다 OS를 설치하는 가상머신과 세부 구조가 다릅니다. 호스트 OS의 리눅스 커널을 활용하는 도커 엔진 위에서 동작하여 게스트 머신마다 OS를 설치 할 필요가 없는 것이 장점입니다.

도커 설치의 리눅스와 윈도우에서의 설치 방법이 다릅니다. 이 문서는 두 환경 모두의 설치가이드를 준비했습니다.

1.1. 도커 엔진과 도커 데스크탑의 차이

도커 엔진은 애플리케이션을 빌드하고 컨테이너화하기 위한 오픈 소스 컨테이너화 기술입니다. 도커 데스크탑은 도커 엔진에 컨테이너, 이미지를 관리 할 수 있는 기능과 이러저러한 편의기능을 덧붙인 프로그램입니다.

2. 시작하기

2.1. 리눅스 환경에서 도커엔진 설치하기

2.1.1. AWS EC2 환경에서 도커 설치하기

이 문서에선 AWS의 EC2 서버에 우분투 **CLI** 환경을 구성 한 뒤 도커를 설치하였습니다.

먼저 `sudo apt-get update`를 입력하여 패키지 툴을 업데이트 한 뒤 진행합니다.

그리고 다음 명령어를 입력하여 도커 레포지토리를 설치 합니다.

```
sudo apt-get install \  
    ca-certificates \  
    curl \  
    gnupg \  
    lsb-release
```

```
sudo mkdir -p /etc/apt/keyrings  
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o  
/etc/apt/keyrings/docker.gpg
```

```
echo \  
    "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]  
    https://download.docker.com/linux/ubuntu \  
    $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list >  
/dev/null
```

```
sudo apt-get update
```

명령어를 입력하면 다음 이미지와 같이 진행됩니다.

```
Reading package lists... Done
E: The repository 'http://ppa.launchpad.net/certbot/certbot/ubuntu focal Release
' does not have a Release file.
N: Updating from such a repository can't be done securely, and is therefore disa
bled by default.
N: See apt-secure(8) manpage for repository creation and user configuration deta
ils.
ubuntu@ip-172-26-9-33:~$
ubuntu@ip-172-26-9-33:~$
ubuntu@ip-172-26-9-33:~$
ubuntu@ip-172-26-9-33:~$ sudo apt-get install \
    ca-certificates \
    curl \
    gnupg \
    lsb-release
Reading package lists... Done
Building dependency tree
Reading state information... Done
lsb-release is already the newest version (11.1.0ubuntu2).
ca-certificates is already the newest version (20211016ubuntu0.20.04.1).
ca-certificates set to manually installed.
curl is already the newest version (7.68.0-1ubuntu2.15).
curl set to manually installed.
gnupg is already the newest version (2.2.19-3ubuntu2.2).
gnupg set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 27 not upgraded.
ubuntu@ip-172-26-9-33:~$
ubuntu@ip-172-26-9-33:~$
ubuntu@ip-172-26-9-33:~$
```

```

ubuntu@ip-172-26-9-33:~$ sudo mkdir -p /etc/apt/keyrings
ubuntu@ip-172-26-9-33:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg
| sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
ubuntu@ip-172-26-9-33:~$ echo \
> "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.g
pg] https://download.docker.com/linux/ubuntu \
> $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list >
/dev/null
ubuntu@ip-172-26-9-33:~$ sudo apt-get update
Hit:1 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu focal-updates InReleas
e
Hit:3 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu focal-backports InRele
ase
Get:4 https://download.docker.com/linux/ubuntu focal InRelease [57.7 kB]
Ign:5 https://pkg.jenkins.io/debian-stable binary/ InRelease
Hit:6 https://pkg.jenkins.io/debian-stable binary/ Release
Hit:7 https://deb.nodesource.com/node_16.x focal InRelease
Hit:8 http://security.ubuntu.com/ubuntu focal-security InRelease
Get:9 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages [22.5
kB]
Ign:11 http://ppa.launchpad.net/certbot/certbot/ubuntu focal InRelease
Err:12 http://ppa.launchpad.net/certbot/certbot/ubuntu focal Release
404 Not Found [IP: 185.125.190.52 80]
Reading package lists... Done
E: The repository 'http://ppa.launchpad.net/certbot/certbot/ubuntu focal Release
' does not have a Release file.
N: Updating from such a repository can't be done securely, and is therefore disa
bled by default.
N: See apt-secure(8) manpage for repository creation and user configuration deta

```

`sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin`를 입력하면 다음 이미지 처럼 도커엔진이 설치됩니다.

```

ubuntu@ip-172-26-9-33:~$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  docker-ce-rootless-extras docker-scan-plugin pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-ce docker-ce-cli docker-ce-rootless-extras
  docker-compose-plugin docker-scan-plugin pigz slirp4netns
0 upgraded, 8 newly installed, 0 to remove and 27 not upgraded.
Need to get 111 MB of archives.
After this operation, 428 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 pigz amd64 2.4-1 [57.4 kB]
Get:2 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 slirp4netns amd64 0.4.3-1 [74.3 kB]
Get:3 https://download.docker.com/linux/ubuntu focal/stable amd64 containerd.io amd64 1.6.15-1 [27.7 MB]
Get:4 https://download.docker.com/linux/ubuntu focal/stable amd64 docker-ce-cli amd64 5:20.10.22~3-0~ubuntu-focal [41.5 MB]
Get:5 https://download.docker.com/linux/ubuntu focal/stable amd64 docker-ce amd64 5:20.10.22~3-0~ubuntu-focal [20.5 MB]
Get:6 https://download.docker.com/linux/ubuntu focal/stable amd64 docker-ce-rootless-extras amd64 5:20.10.22~3-0~ubuntu-focal [8395 kB]
Get:7 https://download.docker.com/linux/ubuntu focal/stable amd64 docker-compose-plugin amd64 2.14.1~ubuntu-focal [9562 kB]

```

`docker version`을 입력하면 도커가 설치되어 있다는 것을 확인 할 수 있습니다.

```

ubuntu@ip-172-26-9-33:~$ docker version
Client: Docker Engine - Community
 Version:           20.10.22
 API version:       1.41
 Go version:        go1.18.9
 Git commit:        3a2c30b
 Built:             Thu Dec 15 22:28:08 2022
 OS/Arch:           linux/amd64
 Context:           default
 Experimental:      true
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.24/version": dial unix /var/run/docker.sock: connect: permission denied
ubuntu@ip-172-26-9-33:~$

```

`systemctl status docker.service`를 입력하여 도커가 실행 되고 있다는 점을 확인하고 `docker run` 명령어를 이용해 도커 이미지 실행을 해보겠습니다. 예제로 hello-world 이미지를 구동합니다. 결과는 다음 이미지와 같습니다.

```
ubuntu@ip-172-26-9-33:~$ systemctl status docker.service
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset:
   Active: active (running) since Tue 2023-01-17 08:30:49 UTC; 5min ago
   TriggeredBy: ● docker.socket
     Docs: https://docs.docker.com
    Main PID: 1074440 (dockerd)
       Tasks: 7
      Memory: 20.9M
     CGroup: /system.slice/docker.service
             └─1074440 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/c
Jan 17 08:30:48 ip-172-26-9-33 dockerd[1074440]: time="2023-01-17T08:30:48.7788>
Jan 17 08:30:48 ip-172-26-9-33 dockerd[1074440]: time="2023-01-17T08:30:48.7788>
Jan 17 08:30:48 ip-172-26-9-33 dockerd[1074440]: time="2023-01-17T08:30:48.7788>
Jan 17 08:30:48 ip-172-26-9-33 dockerd[1074440]: time="2023-01-17T08:30:48.7793>
Jan 17 08:30:48 ip-172-26-9-33 dockerd[1074440]: time="2023-01-17T08:30:48.9833>
Jan 17 08:30:49 ip-172-26-9-33 dockerd[1074440]: time="2023-01-17T08:30:49.0550>
Jan 17 08:30:49 ip-172-26-9-33 dockerd[1074440]: time="2023-01-17T08:30:49.0839>
Jan 17 08:30:49 ip-172-26-9-33 dockerd[1074440]: time="2023-01-17T08:30:49.0841>
Jan 17 08:30:49 ip-172-26-9-33 systemd[1]: Started Docker Application Container
Jan 17 08:30:49 ip-172-26-9-33 dockerd[1074440]: time="2023-01-17T08:30:49.1297>
lines 1-21/21 (END)
```

```
ubuntu@ip-172-26-9-33:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:aa0cc8055b82dc2509bed2e19b275c8f463506616377219d9642221ab53cf9fe
Status: Downloaded newer image for hello-world:latest
```

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/>

```
ubuntu@ip-172-26-9-33:~$
```

2.1.2. VMware를 이용한 리눅스 가상환경

이 문서에선 VMware 가상머신에 우분투 **GUI** 환경을 설치하여 도커를 설치 하였습니다.

진행 과정은 2.1.1. 의 EC2에서의 과정과 매우 유사합니다.

리눅스 터미널창에서 다음 명령어를 입력합니다.

```
sudo apt-get update
sudo apt-get install \
    ca-certificates \
    curl \
    gnupg \
    lsb-release
```

```
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/etc/apt/keyrings/docker.gpg
```

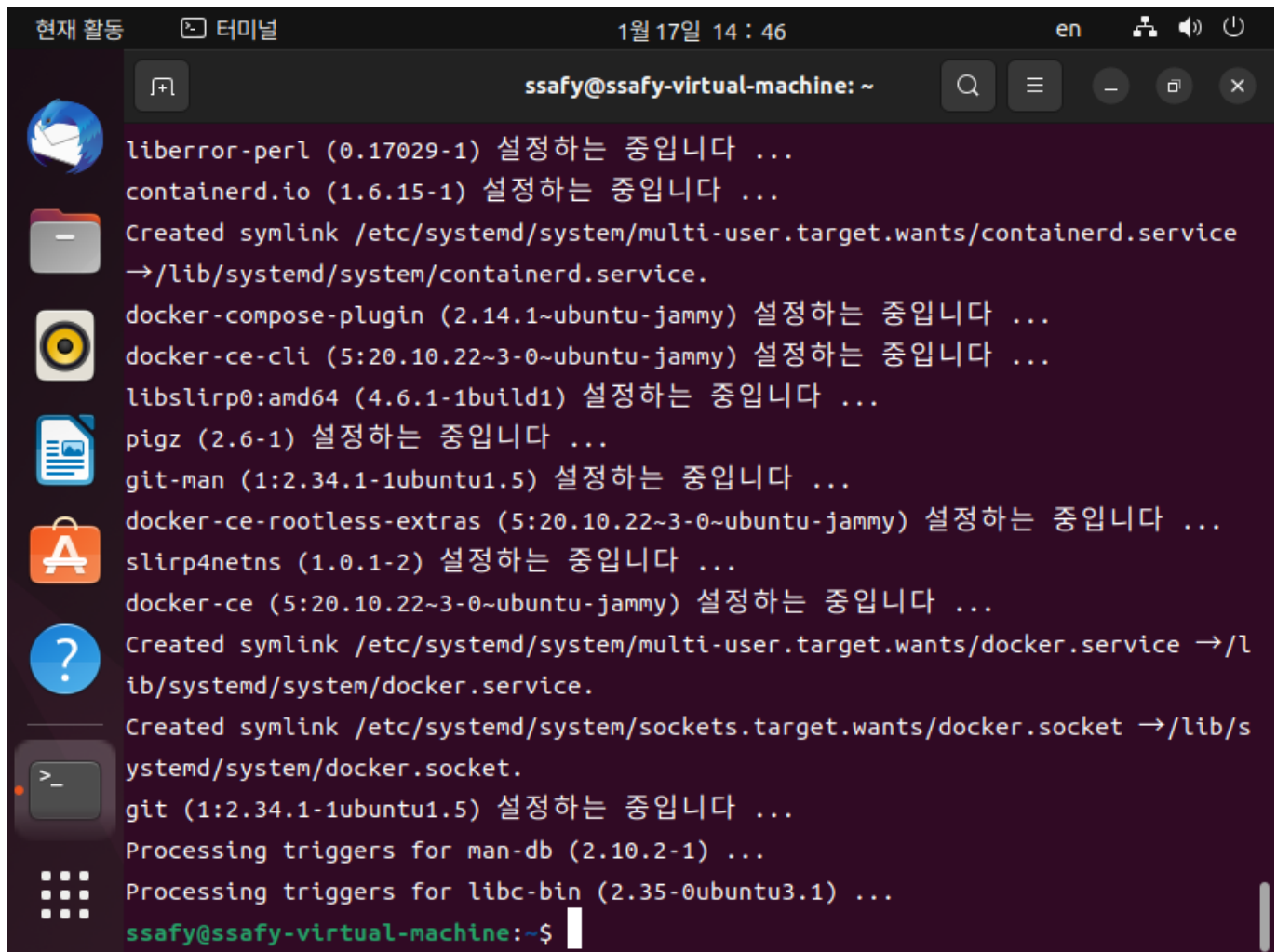
```
echo \
    "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu \
    $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list >
/dev/null
```

위 명령어들로 도커 레포지토리를 설치 할 수 있습니다.

그리고 도커 설치를 위해 다음 명령어를 입력합니다.

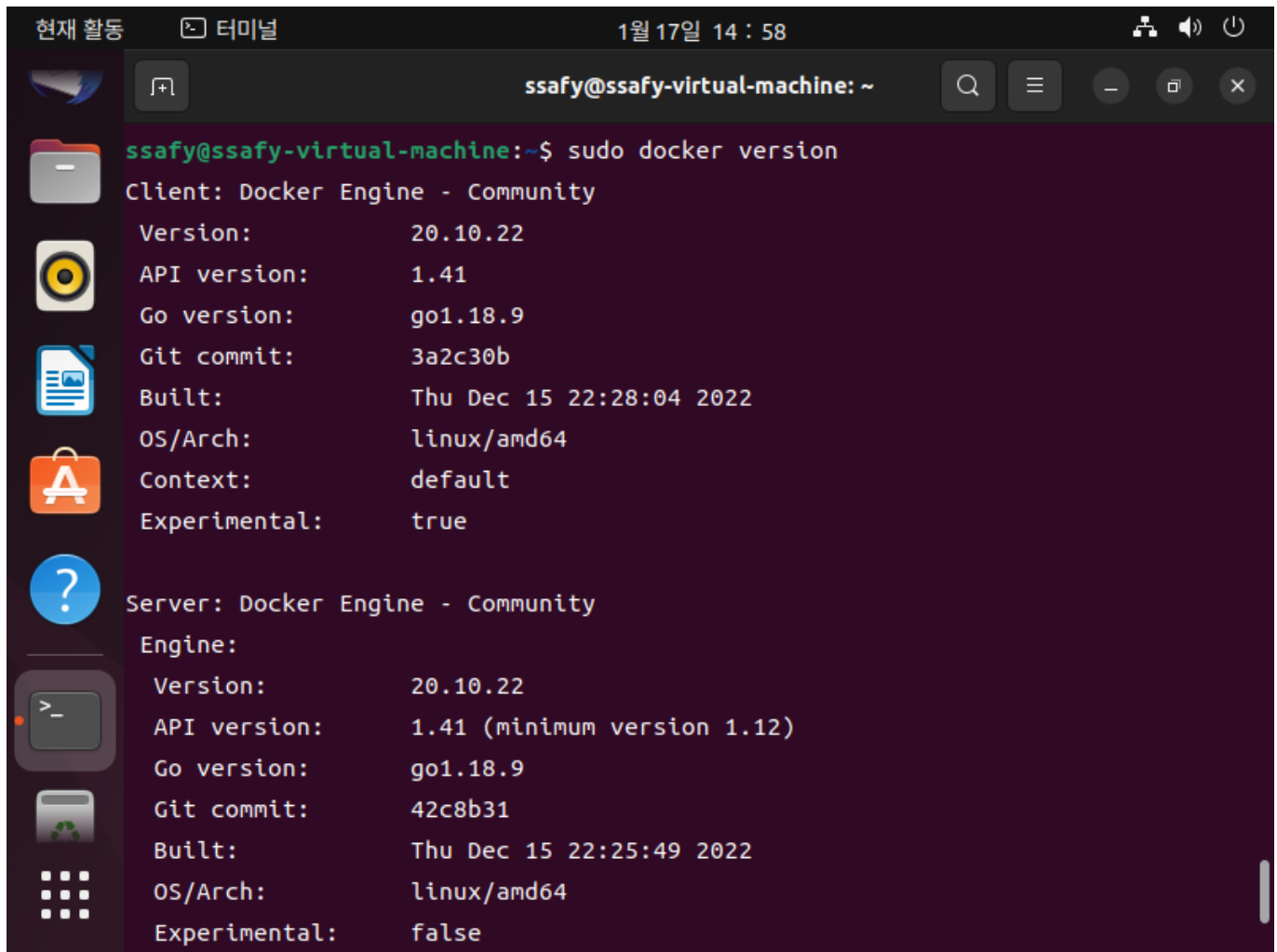
```
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin
```

그러면 다음과 같이 설치가 진행됩니다.



```
현재 활동  터미널 1월 17일 14 : 46 en
ssafy@ssafy-virtual-machine: ~
liberror-perl (0.17029-1) 설정하는 중입니다 ...
containerd.io (1.6.15-1) 설정하는 중입니다 ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service
→/lib/systemd/system/containerd.service.
docker-compose-plugin (2.14.1~ubuntu-jammy) 설정하는 중입니다 ...
docker-ce-cli (5:20.10.22~3-0~ubuntu-jammy) 설정하는 중입니다 ...
libslirp0:amd64 (4.6.1-1build1) 설정하는 중입니다 ...
pigz (2.6-1) 설정하는 중입니다 ...
git-man (1:2.34.1-1ubuntu1.5) 설정하는 중입니다 ...
docker-ce-rootless-extras (5:20.10.22~3-0~ubuntu-jammy) 설정하는 중입니다 ...
slirp4netns (1.0.1-2) 설정하는 중입니다 ...
docker-ce (5:20.10.22~3-0~ubuntu-jammy) 설정하는 중입니다 ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service →/l
ib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket →/lib/s
ystemd/system/docker.socket.
git (1:2.34.1-1ubuntu1.5) 설정하는 중입니다 ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.1) ...
ssafy@ssafy-virtual-machine:~$
```

`sudo docker version`을 입력하여 도커가 설치된 것을 확인 할 수 있습니다.

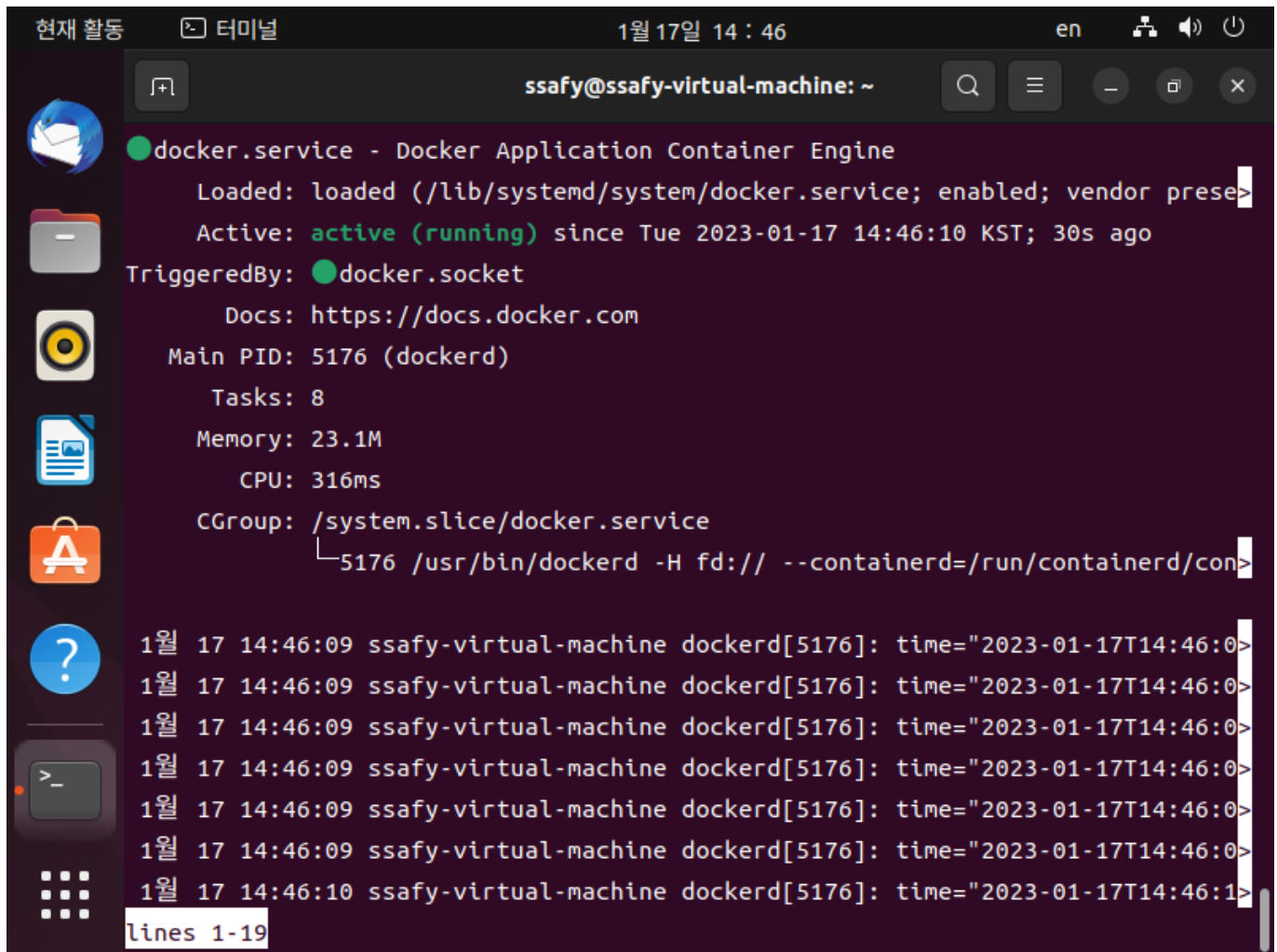


The image shows a terminal window titled "현재 활동 터미널" (Current Activity Terminal) with a timestamp of "1월 17일 14 : 58". The terminal prompt is "ssafy@ssafy-virtual-machine: ~". The user has executed the command "sudo docker version". The output displays the Docker Client and Server versions and their configurations.

```
ssafy@ssafy-virtual-machine:~$ sudo docker version
Client: Docker Engine - Community
 Version:           20.10.22
 API version:       1.41
 Go version:        go1.18.9
 Git commit:        3a2c30b
 Built:             Thu Dec 15 22:28:04 2022
 OS/Arch:           linux/amd64
 Context:           default
 Experimental:       true

Server: Docker Engine - Community
 Engine:
  Version:           20.10.22
  API version:       1.41 (minimum version 1.12)
  Go version:        go1.18.9
  Git commit:        42c8b31
  Built:             Thu Dec 15 22:25:49 2022
  OS/Arch:           linux/amd64
  Experimental:       false
```

`systemctl status docker.service`를 입력하여 도커가 실행중인지 확인합니다.

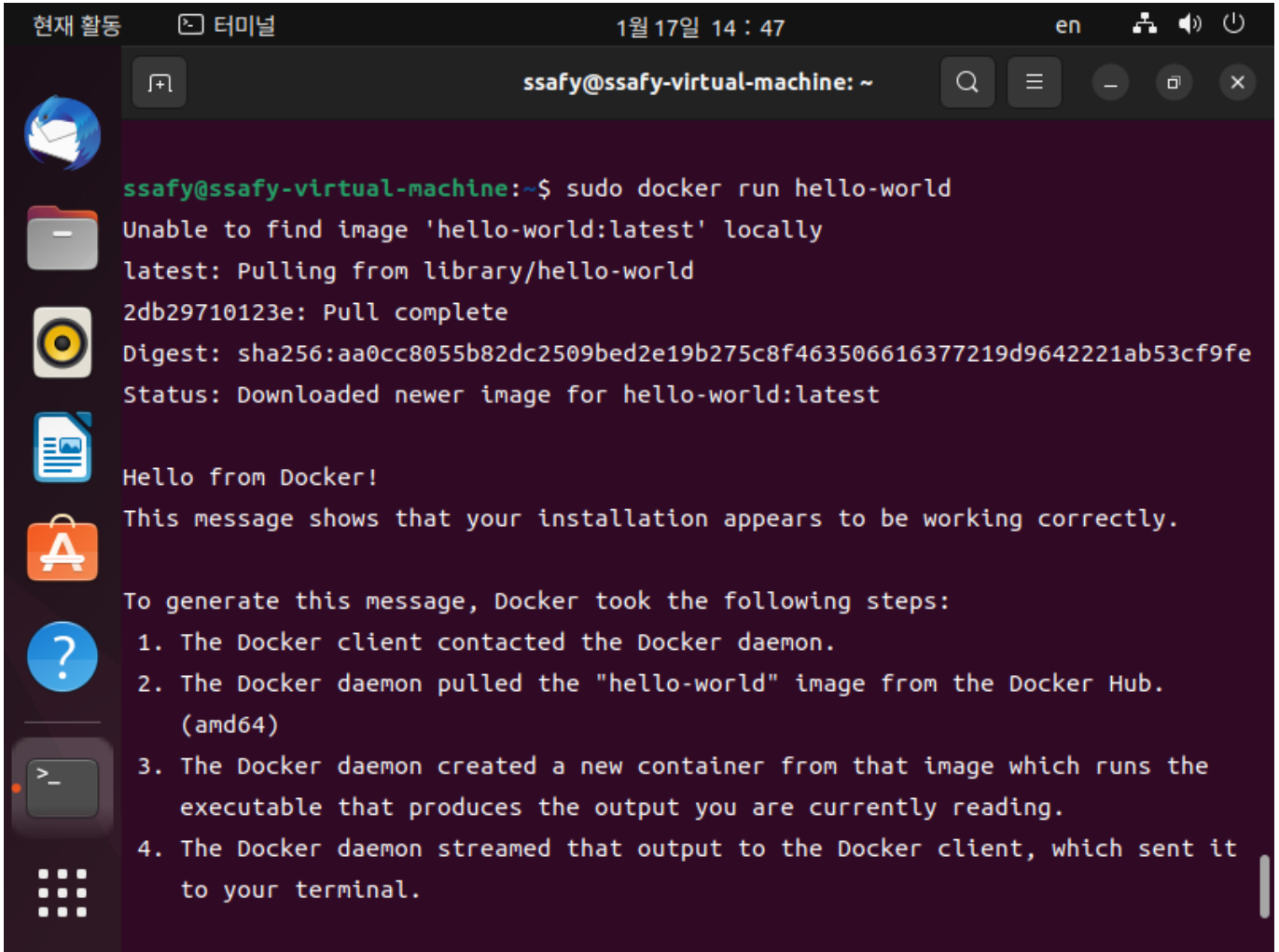
A terminal window titled 'ssafy@ssafy-virtual-machine: ~' showing the status of the Docker service. The service is 'active (running)' and has been running since 14:46:10 KST on 2023-01-17. It shows details like Main PID (5176), Tasks (8), Memory (23.1M), CPU (316ms), and CGroup. Below this, there are several log lines from the dockerd process, all showing 'time="2023-01-17T14:46:09"' and 'time="2023-01-17T14:46:10"'. The terminal also shows a sidebar with various application icons and a top bar with system status and window controls.

```
ssafy@ssafy-virtual-machine: ~  
● docker.service - Docker Application Container Engine  
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor prese  
   Active: active (running) since Tue 2023-01-17 14:46:10 KST; 30s ago  
   TriggeredBy: ● docker.socket  
     Docs: https://docs.docker.com  
    Main PID: 5176 (dockerd)  
      Tasks: 8  
    Memory: 23.1M  
       CPU: 316ms  
    CGroup: /system.slice/docker.service  
            └─5176 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/con>  
  
1월 17 14:46:09 ssafy-virtual-machine dockerd[5176]: time="2023-01-17T14:46:09>  
1월 17 14:46:09 ssafy-virtual-machine dockerd[5176]: time="2023-01-17T14:46:09>  
1월 17 14:46:09 ssafy-virtual-machine dockerd[5176]: time="2023-01-17T14:46:09>  
1월 17 14:46:09 ssafy-virtual-machine dockerd[5176]: time="2023-01-17T14:46:09>  
1월 17 14:46:09 ssafy-virtual-machine dockerd[5176]: time="2023-01-17T14:46:09>  
1월 17 14:46:09 ssafy-virtual-machine dockerd[5176]: time="2023-01-17T14:46:09>  
1월 17 14:46:09 ssafy-virtual-machine dockerd[5176]: time="2023-01-17T14:46:09>  
1월 17 14:46:10 ssafy-virtual-machine dockerd[5176]: time="2023-01-17T14:46:10>  
lines 1-19
```

만약 위처럼 도커가 실행중이 아니라면

`systemctl start docker.service`를 입력하여 도커를 실행시킵니다.

`sudo docker run hello-world`를 입력하여 도커 이미지를 실행을 테스트합니다.



```

현재 활동  터미널 1월 17일 14 : 47 en
ssafy@ssafy-virtual-machine: ~
ssafy@ssafy-virtual-machine:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:aa0cc8055b82dc2509bed2e19b275c8f463506616377219d9642221ab53cf9fe
Status: Downloaded newer image for hello-world:latest

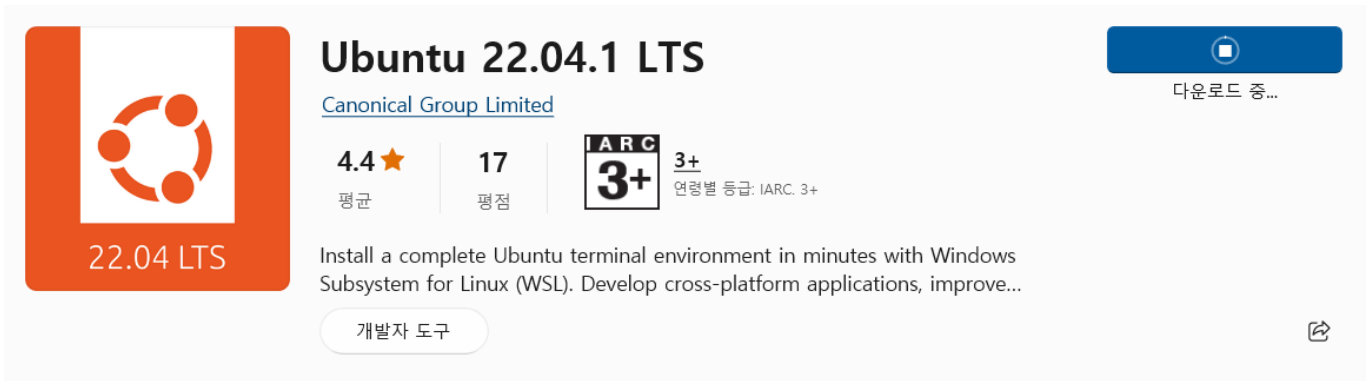
Hello from Docker!
This message shows that your installation appears to be working correctly.

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.

```

2.2. 윈도우 환경에서 WSL2을 활용하여 설치하기

도커는 리눅스 환경에서 동작하기 때문에 윈도우 하위 리눅스 시스템인 WSL2를 활용하여 설치할 수 있습니다. WSL2 환경 셋팅이 되어있다는 전제하에 진행하겠습니다. 이 문서에선 우분투 22.04.1 LTS 버전을 선택하여 진행하겠습니다.



Ubuntu 22.04.1 LTS
 Canonical Group Limited

4.4 ★ 평균 | 17 평점 | ARC 3+ | 3+ 연령별 등급: IARC: 3+

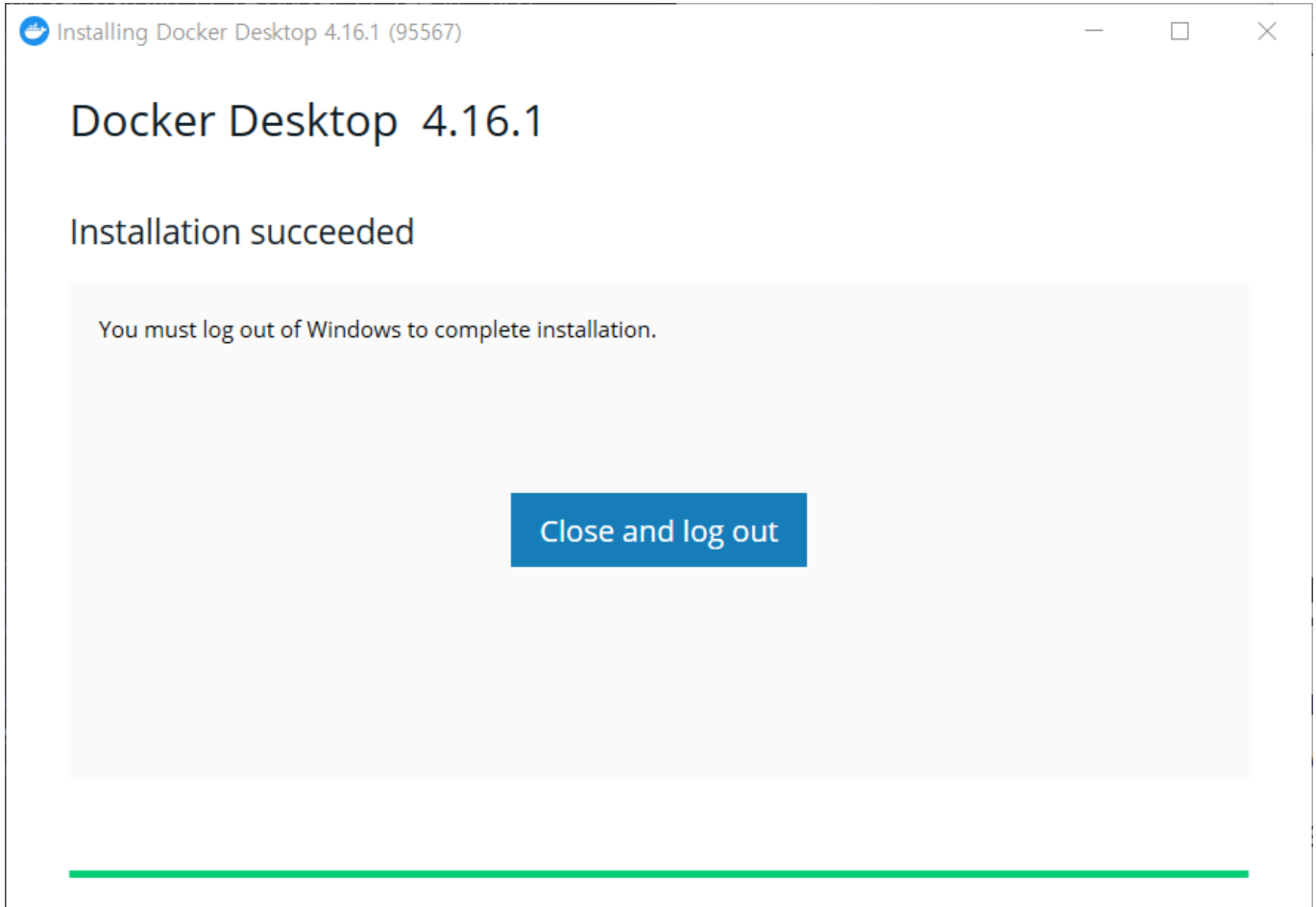
Install a complete Ubuntu terminal environment in minutes with Windows Subsystem for Linux (WSL). Develop cross-platform applications, improve...

개발자 도구

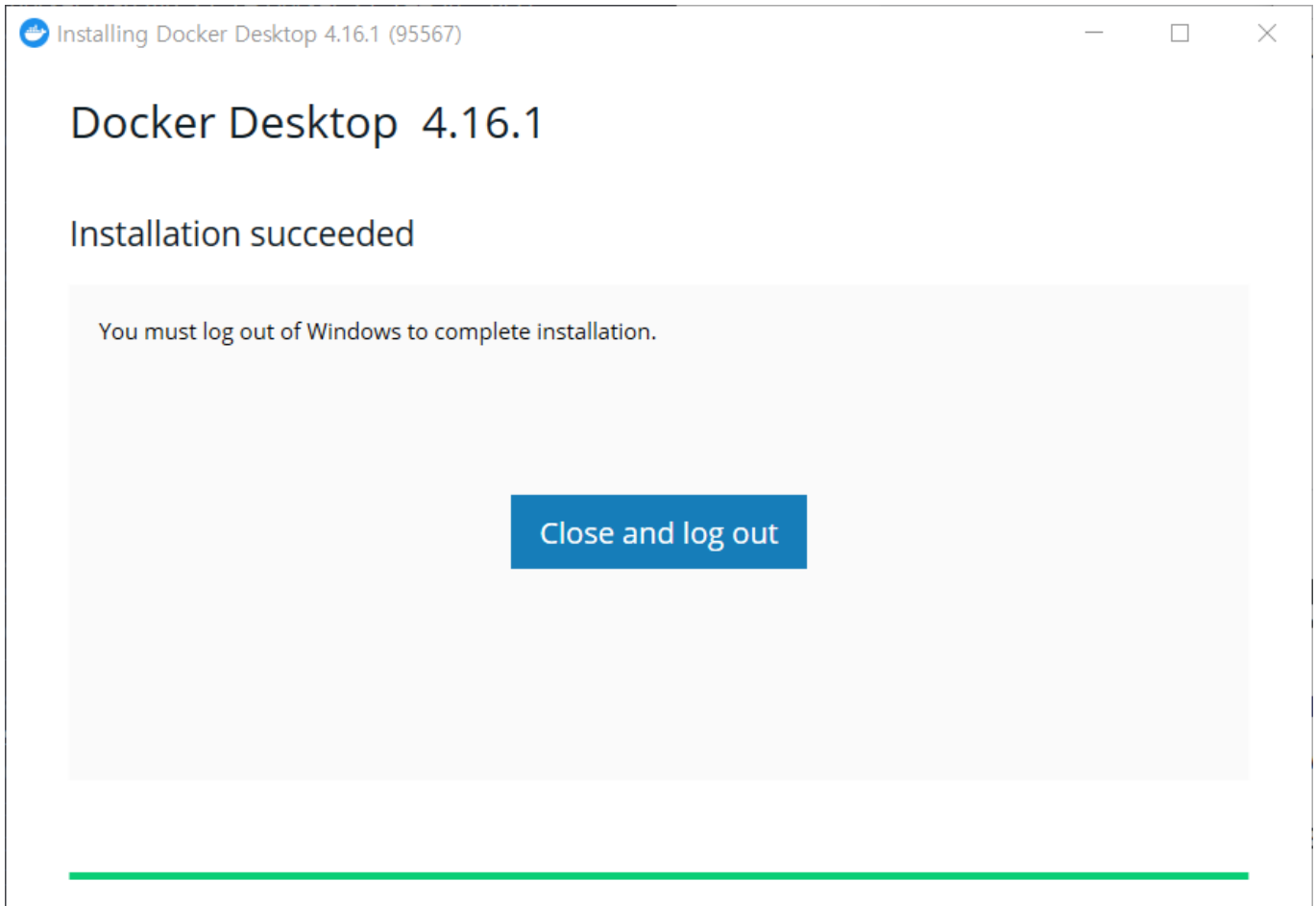
다운로드 중...

다음은 윈도우 환경에서 도커 데스크탑을 [다운로드](#) 합니다.

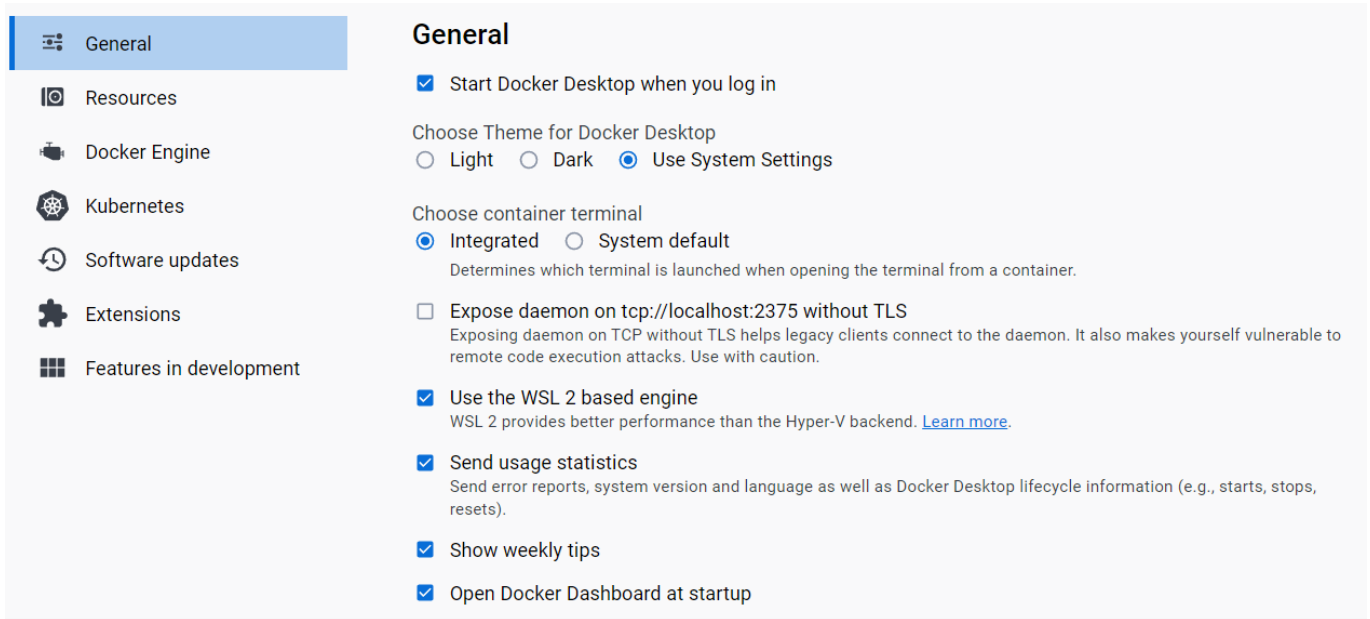
설치 도중 WSL2 기반 엔진 사용 확인란을 선택하고 설치를 진행합니다.



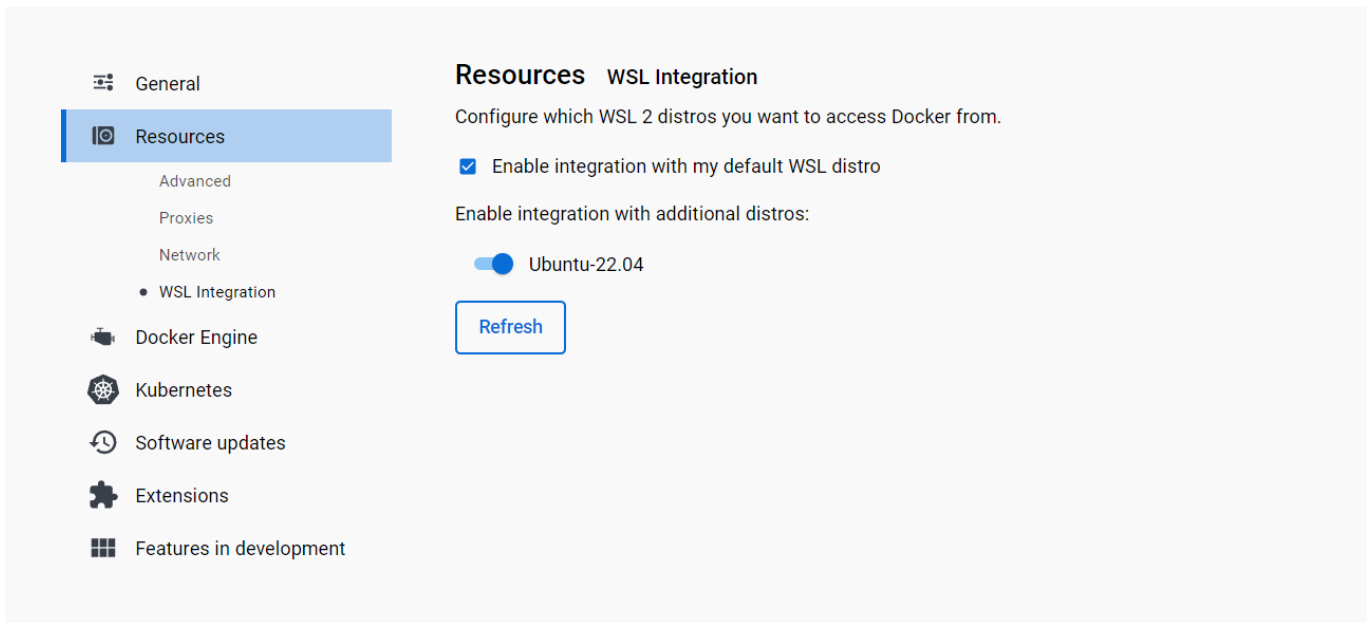
설치가 되면 PC 재부팅을 하게 됩니다.



설정 일반 탭에서 Use the WSL 2 based engine이 체크 되어있는지 확인합니다.



설정 리소스 탭에서 WSL 로 설치한 이미지를 골라 체크 합니다.



주의! 도커 데스크탑은 Hyper-V 가상환경을 사용하기 때문에 VMware나 VirtualBox가 같이 사용되지 않습니다.

우분투 환경에서 `sudo docker version`을 입력하여 설치가 되어있음을 확인합니다.

```
ssafy@DESKTOP-KVCQHCD: ~  
ssafy@DESKTOP-KVCQHCD:~$ docker version  
Client: Docker Engine - Community  
Cloud integration: v1.0.29  
Version: 20.10.22  
API version: 1.41  
Go version: go1.18.9  
Git commit: 3a2c30b  
Built: Thu Dec 15 22:28:22 2022  
OS/Arch: linux/amd64  
Context: default  
Experimental: true  
  
Server: Docker Desktop  
Engine:  
Version: 20.10.22  
API version: 1.41 (minimum version 1.12)  
Go version: go1.18.9  
Git commit: 42c8b31  
Built: Thu Dec 15 22:26:14 2022  
OS/Arch: linux/amd64  
Experimental: false  
containerd:  
Version: 1.6.14  
GitCommit: 9ba4b250366a5ddde94bb7c9d1def331423aa323  
runc:  
Version: 1.1.4  
GitCommit: v1.1.4-0-g5fd4c4d  
docker-init:  
Version: 0.19.0  
GitCommit: de40ad0  
ssafy@DESKTOP-KVCQHCD:~$
```

`sudo docker run hello-world`를 입력하면 hello world 컨테이너가 생성되고 실행 됨을 확인 할 수 있습니다.

```
ssafy@DESKTOP-KVCQHCD: ~  
ssafy@DESKTOP-KVCQHCD:~$ sudo docker run hello-world  
[sudo] password for ssafy:  
Unable to find image 'hello-world:latest' locally  
latest: Pulling from library/hello-world  
2db29710123e: Pull complete  
Digest: sha256:aa0cc8055b82dc2509bed2e19b275c8f463506616377219d9642221ab53cf9fe  
Status: Downloaded newer image for hello-world:latest  
  
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/  
ssafy@DESKTOP-KVCQHCD:~$
```

3. 주요 명령어

- `dockser create` : 새 컨테이너를 만듭니다.
- `dockser run` : 새 컨테이너를 실행시킵니다.
- `dockser start` : 정지한 컨테이너를 실행시킵니다.

- `dockser restart` : 컨테이너를 재시작 합니다.
- `dockser pause` : 컨테이너의 프로세스를 일시중지합니다.
- `dockser stop` : 컨테이너를 종료합니다.
- `dockser kill` : 컨테이너를 강제종료합니다.
- `dockser rm` : 컨테이너를 삭제합니다.
- `dockser unpause` : 일시정지한 컨테이너를 다시 동작합니다.
- `dockser images` : 이미지 리스트를 출력합니다.
- `dockser history` : 이미지의 과거 이력을 출력합니다.
- `dockser update` : 컨테이너 구성을 업데이트합니다.
- `dockser rename` : 컨테이너의 이름을 재지정합니다.
- `dockser logs` : 컨테이너의 로그를 출력합니다.
- `dockser ps` : 컨테이너 리스트를 나열합니다.
- `dockser container ls` : 컨테이너 리스트를 나열합니다.
- `dockser version` : 도커의 버전 정보를 표시합니다.