

WSL + docker + GPU (4)

そろそろ環境構築を終わらせたいので、一度いままでの内容をまとめようと思う。

参考

- [nvidia-docker コンテナ Deep Learning](#)
- [nvidia docker って今どうなってるの? \(19.11 版\)](#)
- [nvidia docker って今どうなってるの? \(20.09 版\)](#)

1. 環境構築

1.1. Ubuntu の初期設定

まずパッケージのアップデートを行う。

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

1.2. WSL kernel update

WSL2 のカーネル **4.19.121-microsoft** 以上で動作しているかを確認する。

```
uname -r
```

```
kohei@DESKTOP-Q123T6P:~$ uname -r  
4.19.128-microsoft-standard
```

もし、**4.19.121-microsoft** より低い version で WSL が動作している場合は、[このページ](#)を参考に、kernel を version up する。

1.3. NVIDIA ドライバのインストール

CUDA Toolkit の [web サイト](#) にアクセスし、自分が欲しいディストリビューションに合わせたダウンロードファイルを選択する。

CUDA Toolkit

Develop, Optimize and Deploy GPU-Accelerated Apps

The NVIDIA® CUDA® Toolkit provides a development environment for creating high performance GPU-accelerated applications. With the CUDA Toolkit, you can develop, optimize, and deploy your applications on GPU-accelerated embedded systems, desktop workstations, enterprise data centers, cloud-based platforms and HPC supercomputers. The toolkit includes GPU-accelerated libraries, debugging and optimization tools, a C/C++ compiler, and a runtime library to deploy your application.

Using built-in capabilities for distributing computations across multi-GPU configurations, scientists and researchers can develop applications that scale from single GPU workstations to cloud installations with thousands of GPUs.

[Download Now](#)

CUDA Toolkit 11.0 Update 1 Downloads

Home > High Performance Computing > CUDA Toolkit > CUDA Toolkit 11.0 Update 1 Downloads

Select Target Platform

Click on the green buttons that describe your target platform. Only supported platforms will be shown. By downloading and using the software, you agree to fully comply with the terms and conditions of the [CUDA EULA](#).

Operating System

[Windows](#)[Linux](#)

Architecture

[x86_64](#)[ppc64le](#)[sbsa](#)

Distribution

[OpenSUSE](#)[RHEL](#)[CentOS](#)[SLES](#)[Ubuntu](#)

Version

[20.04](#)[18.04](#)[16.04](#)

Installer Type

[runfile \(local\)](#)[deb \(local\)](#)[deb \(network\)](#)

Download Installer for Linux Ubuntu 20.04 x86_64

The base installer is available for download below.

Base Installer

Installation Instructions:

```
$ wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/x86_64/cuda-ubuntu2004.pin
$ sudo mv cuda-ubuntu2004.pin /etc/apt/preferences.d/cuda-repository-pin-600
$ wget https://developer.download.nvidia.com/compute/cuda/11.0.3/local_installers/cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb
$ sudo dpkg -i cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb
$ sudo apt-key add /var/cuda-repo-ubuntu2004-11-0-local/7fa2af80.pub
$ sudo apt-get update
$ sudo apt-get -y install cuda
```

ここで、最後の行を

```
sudo apt-get -y install cuda-drivers
```

とすると、最新のドライバだけがきれいにインストールされる。

以下、CUDA Toolkit に記された通りにインストールする。

```
wget
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/x86_64/cuda-ubuntu2004.pin

sudo mv cuda-ubuntu2004.pin /etc/apt/preferences.d/cuda-repository-pin-600

wget
https://developer.download.nvidia.com/compute/cuda/11.0.3/local_installers/cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb

sudo dpkg -i cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb

sudo apt-key add /var/cuda-repo-ubuntu2004-11-0-local/7fa2af80.pub

sudo apt update

sudo apt -y install cuda-drivers
```

```
kohei@DESKTOP-Q123T6P:~$ wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/x86_64/cuda-ubuntu2004.pin
--2020-09-12 11:54:33-- https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/x86_64/cuda-ubuntu2004.pin
Resolving developer.download.nvidia.com (developer.download.nvidia.com)... 152.199.39.144
Connecting to developer.download.nvidia.com (developer.download.nvidia.com)[152.199.39.144]:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 190 [application/octet-stream]
Saving to: 'cuda-ubuntu2004.pin'

cuda-ubuntu2004.pin 100%[=====] 190 --KB/s in 0s

2020-09-12 11:54:33 (11.4 MB/s) - 'cuda-ubuntu2004.pin' saved [190/190]

kohei@DESKTOP-Q123T6P:~$ sudo mv cuda-ubuntu2004.pin /etc/apt/preferences.d/cuda-repository-pin-600
kohei@DESKTOP-Q123T6P:~$ wget https://developer.download.nvidia.com/compute/cuda/11.0.3/local_installers/cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb
--2020-09-12 11:55:54-- https://developer.download.nvidia.com/compute/cuda/11.0.3/local_installers/cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb
Resolving developer.download.nvidia.com (developer.download.nvidia.com)... 152.199.39.144
Connecting to developer.download.nvidia.com (developer.download.nvidia.com)[152.199.39.144]:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2306981744 (2.1G) [application/x-deb]
Saving to: 'cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb'

cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb 100%[=====] 2.15G 5.00MB/s in 11m 38s

2020-09-12 12:07:32 (3.15 MB/s) - 'cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb' saved [2306981744/2306981744]

kohei@DESKTOP-Q123T6P:~$ sudo dpkg -i cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb
[sudo] password for kohei:
Sorry, try again.
[sudo] password for kohei:
Selecting previously unselected package cuda-repo-ubuntu2004-11-0-local.
(Reading database ... 31912 files and directories currently installed.)
Preparing to unpack cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb ...
Unpacking cuda-repo-ubuntu2004-11-0-local (11.0.3-450.51.06-1) ...
Setting up cuda-repo-ubuntu2004-11-0-local (11.0.3-450.51.06-1) ...

The public CUDA GPG key does not appear to be installed.
To install the key, run this command:
sudo apt-key add /var/cuda-repo-ubuntu2004-11-0-local/7fa2af80.pub

kohei@DESKTOP-Q123T6P:~$ sudo apt-key add /var/cuda-repo-ubuntu2004-11-0-local/7fa2af80.pub
OK
kohei@DESKTOP-Q123T6P:~$
```

```
kohei@DESKTOP-Q123T6P:~$ sudo apt-get update
Get:1 file:/var/cuda-repo-ubuntu2004-11-0-local InRelease
Ign:1 file:/var/cuda-repo-ubuntu2004-11-0-local InRelease
Get:2 file:/var/cuda-repo-ubuntu2004-11-0-local Release [564 B]
Get:2 file:/var/cuda-repo-ubuntu2004-11-0-local Release [564 B]
Get:3 file:/var/cuda-repo-ubuntu2004-11-0-local Release.gpg [836 B]
Get:3 file:/var/cuda-repo-ubuntu2004-11-0-local Release.gpg [836 B]
Get:4 file:/var/cuda-repo-ubuntu2004-11-0-local Packages [24.4 kB]
Hit:5 http://archive.ubuntu.com/ubuntu focal InRelease
Get:6 http://archive.ubuntu.com/ubuntu focal-updates InRelease [111 kB]
Get:7 http://security.ubuntu.com/ubuntu focal-security InRelease [107 kB]
Get:8 http://archive.ubuntu.com/ubuntu focal-backports InRelease [98.3 kB]
Get:9 http://archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [165 kB]
Fetched 481 kB in 3s (187 kB/s)
Reading package lists... Done
kohei@DESKTOP-Q123T6P:~$
```

1.4. Docker のインストール

Docker [公式のサイト](#)の手順に従って、Docker をインストールする。

1. まず、古い version の Docker をアンインストールする。

```
sudo apt remove docker docker-engine docker.io containerd runc
```

2. 以下のコマンドで必要なリポジトリを追加する。

```
sudo apt update

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

```
kohei@DESKTOP-Q123T6P:~$ sudo apt install \
> apt-transport-https \
> ca-certificates \
> curl \
> gnupg-agent \
> software-properties-common
Reading package lists... Done
Building dependency tree
Reading state information... Done
ca-certificates is already the newest version (20190110ubuntu1.1).
ca-certificates set to manually installed.
curl is already the newest version (7.68.0-1ubuntu2.2).
curl set to manually installed.
software-properties-common is already the newest version (0.98.9.2).
software-properties-common set to manually installed.
The following NEW packages will be installed:
  apt-transport-https gnupg-agent
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 6944 B of archives.
After this operation, 206 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu focal-updates/universe amd64 apt-transport-https all 2.0.2ubuntu0.1 [1708 B]
Get:2 http://archive.ubuntu.com/ubuntu focal/universe amd64 gnupg-agent all 2.2.19-3ubuntu2 [5236 B]
Fetched 6944 B in 1s (6589 B/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 93373 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_2.0.2ubuntu0.1_all.deb ...
Unpacking apt-transport-https (2.0.2ubuntu0.1) ...
Selecting previously unselected package gnupg-agent.
Preparing to unpack .../gnupg-agent_2.2.19-3ubuntu2_all.deb ...
Unpacking gnupg-agent (2.2.19-3ubuntu2) ...
Setting up apt-transport-https (2.0.2ubuntu0.1) ...
Setting up gnupg-agent (2.2.19-3ubuntu2) ...
kohei@DESKTOP-Q123T6P:~$
```

3. Docker の official GPG key を追加する。

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

4. fingerprint の最後の 8 文字を検索して、手元の key が 9DC8 5822 9FC7 DD38 854A E2D8 8D81 803C 0EBF CD88 であることを確認する。

```
sudo apt-key fingerprint 0EBFCD88
```

```
kohei@DESKTOP-Q123T6P:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
OK
kohei@DESKTOP-Q123T6P:~$ sudo apt-key fingerprint 0EBFCD88
pub   rsa4096 2017-02-22 [SCEA]
      9DC8 5822 9FC7 DD38 854A  E2D8 8D81 803C 0EBF CD88
uid   [ unknown] Docker Release (CE deb) <docker@docker.com>
sub   rsa4096 2017-02-22 [S]
kohei@DESKTOP-Q123T6P:~$
```

5. 次のコマンドを使用して、安定したリポジトリを追加する。nightly または test リポジトリを追加するには、以下のコマンドで stable という単語の後に nightly または test (または両方) という単語を追加する。

今回は、x86_64/amd64 を使用する。

```
x86_64 / amd64  armhf  arm64

$ sudo add-apt-repository \
  "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) \
  stable"
```

```
sudo add-apt-repository \
  "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) \
  stable"
```

```
kohei@DESKTOP-Q123T6P:~$ sudo add-apt-repository \
> "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
> $(lsb_release -cs) \
> stable"
Get:1 file:/var/cuda-repo-ubuntu2004-11-0-local InRelease
Ign:1 file:/var/cuda-repo-ubuntu2004-11-0-local InRelease
Get:2 file:/var/cuda-repo-ubuntu2004-11-0-local Release [564 B]
Get:2 file:/var/cuda-repo-ubuntu2004-11-0-local Release [564 B]
Get:4 https://download.docker.com/linux/ubuntu focal InRelease [36.2 kB]
Get:5 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages [3056 B]
Hit:6 http://security.ubuntu.com/ubuntu focal-security InRelease
Hit:7 http://archive.ubuntu.com/ubuntu focal InRelease
Hit:8 http://archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:9 http://archive.ubuntu.com/ubuntu focal-backports InRelease
Fetched 39.2 kB in 1s (33.2 kB/s)
Reading package lists... Done
```

6. apt パッケージインデックスを更新し、Docker Engine と containerd の最新バージョンをインストールするか、7. の手順に進んで特定のバージョンをインストールする。

```
sudo apt update
```

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

先ほどのコマンドを実行すると、次のような画面が表示された。

```
Configuring grub-pc
The grub-pc package is being upgraded. This menu allows you to select which devices you'd like grub-install
to be automatically run for, if any.

Running grub-install automatically is recommended in most situations, to prevent the installed GRUB core
image from getting out of sync with GRUB modules or grub.cfg.

If you're unsure which drive is designated as boot drive by your BIOS, it is often a good idea to install
GRUB to all of them.

Note: it is possible to install GRUB to partition boot records as well, and some appropriate partitions are
offered here. However, this forces GRUB to use the blocklist mechanism, which makes it less reliable, and
therefore is not recommended.

GRUB install devices:
[ ] /dev/sda (274877 MB; ???)
[ ] /dev/sdb (274877 MB; ???)
[ ] /dev/sdb (274877 MB; ???)

<Ok>
```

```
Configuring grub-pc
You chose not to install GRUB to any devices. If you continue, the boot loader may not be properly
configured, and when this computer next starts up it will use whatever was previously in the boot sector. If
there is an earlier version of GRUB 2 in the boot sector, it may be unable to load modules or handle the
current configuration file.

If you are already using a different boot loader and want to carry on doing so, or if this is a special
environment where you do not need a boot loader, then you should continue anyway. Otherwise, you should
install GRUB somewhere.

Continue without installing GRUB?
<Yes> <No>
```

初めて見るエラーだったので、少し調べてみた。

GRUB

GRUB とは、コンピュータの起動時に最初に読み込まれ、ストレージなどからオペレーティングシステム (OS) を読み込んで起動するブートローダの一つ。GNU プロジェクトが開発・公開しているオープンソースソフトウェアで、よく Linux と組み合わせて用いられる。[e-Words より]

画面については、以下を参考にした。

[Windows10 Update が来たので WSL2 で Ubuntu 20.04 LTS を起動し Docker をインストール](#)

7. Docker Engine の特定のバージョンをインストールするには、利用可能なバージョンをリポジトリにリストしてから、選択してインストールします。

- まず、リポジトリで利用可能なバージョン一覧を表示する。

```
apt-cache madison docker-ce
```

```
kohei@DESKTOP-Q123T6P:~$ apt-cache madison docker-ce
docker-ce | 5:19.03.12~3-0~ubuntu-focal | https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
docker-ce | 5:19.03.11~3-0~ubuntu-focal | https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
docker-ce | 5:19.03.10~3-0~ubuntu-focal | https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
docker-ce | 5:19.03.9~3-0~ubuntu-focal | https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
```

- 2 番目の列のバージョン文字列 `5:19.03.11~3-0~ubuntu-focal` を使用して、特定のバージョンをインストールする場合は、以下のコマンドの `<VERSION_STRING>` 部分をそのバージョン文字列に置き換えて使用する。

```
sudo apt install docker-ce=<VERSION_STRING> docker-ce-cli=<VERSION_STRING>
containerd.io
```

8. docker イメージを実行するために、docker daemon を起動します。

```
sudo service docker start
```

```
kohei@DESKTOP-Q123T6P:~$ sudo service docker start
* Starting Docker: docker [ OK ]
```

9. hello-world イメージを実行して、Docker エンジンが正しくインストールされていることを確認します。

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

```
kohei@DESKTOP-Q123T6P:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
0e03bdcc26d7: Pull complete
Digest: sha256:4cf9c47f86df71d48364001ede3a4fcd85ae80ce02ebad74156906cafff5378bc
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/
kohei@DESKTOP-Q123T6P:~$
```

10. `hello-world` のメッセージが表示されたことを確認したら、docker グループにユーザを追加しておく。docker はデフォルトだと root 権限がないユーザからの操作を受け付けないが、こうすることで、

`sudo` なしに、`docker run` コマンドを使用できるようになる。

- ユーザ追加前の、`docker` グループを確認する。

```
grep -i docker /etc/group
```

- ユーザを `docker` グループに追加する。ここで、`$USER` は自分がログイン時に使っているユーザ名に変更する必要がある。

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

```
kohei@DESKTOP-Q123T6P:~$ grep -i docker /etc/group
docker:x:999:
kohei@DESKTOP-Q123T6P:~$ sudo usermod -aG docker kohei
kohei@DESKTOP-Q123T6P:~$ grep -i docker /etc/group
docker:x:999:kohei
```

11. グループに入ったことを確認した後、一旦ログアウトを行い、再度ログインする。

```
logout
```

12. `docker` サービスの開始と自動起動設定をする。

```
curl https://get.docker.com | sh

sudo systemctl start docker && sudo systemctl enable docker
```

これを実行すると、最後に次のような文章が表示される。

If you would like to use Docker as a non-root user, you should now consider adding your user to the "docker" group with something like:

```
sudo usermod -aG docker kohei
```

Remember that you will have to log out and back in for this to take effect!

適当訳：

Docker を非 root ユーザーとして使用したい場合は、ここで検討する必要があります。次のようにして、ユーザーを「docker」グループに追加します。

```
sudo usermod -aG docker kohei
```

これを有効にするには、ログアウトしてから再度ログインする必要があることに注意してください。

WARNING: Adding a user to the "docker" group will grant the ability to run containers which can be used to obtain root privileges on the docker host. Refer to <https://docs.docker.com/engine/security/security/#docker-daemon-attack-surface> for more information.

適用訳：

警告：「docker」グループにユーザーを追加すると、Docker ホストで root 権限を取得するために使用できるコンテナを実行する機能が付与されます。詳細については、
<https://docs.docker.com/engine/security/security/#docker-daemon-attack-surface> を参照してください。

```
If you would like to use Docker as a non-root user, you should now consider
adding your user to the "docker" group with something like:

sudo usermod -aG docker kohei

Remember that you will have to log out and back in for this to take effect!

WARNING: Adding a user to the "docker" group will grant the ability to run
containers which can be used to obtain root privileges on the
docker host.
Refer to https://docs.docker.com/engine/security/security/#docker-daemon-attack-surface
for more information.
```

このような警告が表示されたため、一応 docker グループにユーザ登録されているかを確認する。

```
grep -i docker /etc/group
```

```
kohei@DESKTOP-Q123T6P:~$ grep -i docker /etc/group
docker:x:999:kohei
kohei@DESKTOP-Q123T6P:~$
```

しっかりと、登録されたままの状態となっていることを確認。

13. 起動確認

最後に起動確認を行う。

```
docker -ps
```

このようにエラーが出なければ OK

```
kohei@DESKTOP-Q123T6P:~$ docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS          NAMES
kohei@DESKTOP-Q123T6P:~$
```

1.5. NVIDIA Container Toolkit のインストール

nvidia の[公式ドキュメントのインストールガイド](#)に従い、NVIDIA Container Toolkit のインストールを行う。

1. Version 1.0 の `nvidia-docker` がインストールされていないかを確認する。

```
docker volume ls -q -f driver=nvidia-docker | xargs -r -I{} -n1 docker ps -q -a -f
volume={} | xargs -r docker rm -f
```

インストールされていれば、以下のコマンドでアンインストールする。

```
sudo apt purge nvidia-docker
```

一応、確認のため実行してみたが、インストールされていなかったため、エラーが出た。

```
kohei@DESKTOP-Q123T6P:~$ docker volume ls -q -f driver=nvidia-docker | xargs -r -I{} -n1 docker ps -q -a -f volume={} | xargs -r docker rm -f
kohei@DESKTOP-Q123T6P:~$ sudo apt purge nvidia-docker
[sudo] password for kohei:
Reading package lists... Done
Building dependency tree
Reading state information... Done
E: Unable to locate package nvidia-docker
```

2. 安定したリポジトリと GPG キーを設定する。

```
distribution=$(. /etc/os-release;echo $ID$VERSION_ID)

curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-key add -

curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.list | sudo tee /etc/apt/sources.list.d/nvidia-docker.list
```

```
kohei@DESKTOP-Q123T6P:~$ distribution=$(. /etc/os-release;echo $ID$VERSION_ID)
kohei@DESKTOP-Q123T6P:~$ curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-key add -
OK
kohei@DESKTOP-Q123T6P:~$ curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.list | sudo tee /etc/apt/sources.list.d/nvidia-docker.list
#deb https://nvidia.github.io/libnvidia-container/stable/ubuntu18.04/${ARCH} /
#deb https://nvidia.github.io/nvidia-container-runtime/stable/ubuntu18.04/${ARCH} /
#deb https://nvidia.github.io/nvidia-container-runtime/experimental/ubuntu18.04/${ARCH} /
deb https://nvidia.github.io/nvidia-docker/ubuntu18.04/${ARCH} /
kohei@DESKTOP-Q123T6P:~$
```

WSL の CUDA や A100 の新しい MIG 機能などの実験的機能にアクセスするには、実験的ブランチをリポジトリリストに追加できます。

```
curl -s -L https://nvidia.github.io/nvidia-container-runtime/experimental/$distribution/nvidia-container-runtime.list | sudo tee /etc/apt/sources.list.d/nvidia-container-runtime.list
```

```
kohei@DESKTOP-Q123T6P:~$ curl -s -L https://nvidia.github.io/nvidia-container-runtime/experimental/$distribution/nvidia-container-runtime.list | sudo tee /etc/apt/sources.list.d/nvidia-container-runtime.list
deb https://nvidia.github.io/libnvidia-container/experimental/ubuntu18.04/${ARCH} /
deb https://nvidia.github.io/nvidia-container-runtime/experimental/ubuntu18.04/${ARCH} /
kohei@DESKTOP-Q123T6P:~$
```

3. パッケージリストを更新した後、nvidia-docker2 パッケージ (および依存関係) をインストールします。

```
sudo apt update

sudo apt install -y nvidia-docker2
```

```
kohei@DESKTOP-Q123T6P:~$ sudo apt update
Get:1 file:/var/cuda-repo-ubuntu2004-11-0-local InRelease
Ign:1 file:/var/cuda-repo-ubuntu2004-11-0-local InRelease
Get:2 file:/var/cuda-repo-ubuntu2004-11-0-local Release [564 B]
Get:2 file:/var/cuda-repo-ubuntu2004-11-0-local Release [564 B]
Hit:4 http://archive.ubuntu.com/ubuntu focal InRelease
Hit:5 http://security.ubuntu.com/ubuntu focal-security InRelease
Hit:6 http://archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:7 http://archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:8 https://download.docker.com/linux/ubuntu focal InRelease
Get:9 https://nvidia.github.io/libnvidia-container/experimental/ubuntu18.04/amd64 InRelease [1158 B]
Get:10 https://nvidia.github.io/nvidia-container-runtime/experimental/ubuntu18.04/amd64 InRelease [1149 B]
Get:11 https://nvidia.github.io/libnvidia-container/stable/ubuntu18.04/amd64 InRelease [1139 B]
Get:12 https://nvidia.github.io/nvidia-container-runtime/stable/ubuntu18.04/amd64 InRelease [1136 B]
Get:13 https://nvidia.github.io/nvidia-docker/ubuntu18.04/amd64 InRelease [1129 B]
Get:14 https://nvidia.github.io/libnvidia-container/experimental/ubuntu18.04/amd64 Packages [3392 B]
Get:15 https://nvidia.github.io/nvidia-container-runtime/experimental/ubuntu18.04/amd64 Packages [804 B]
Get:16 https://nvidia.github.io/libnvidia-container/stable/ubuntu18.04/amd64 Packages [8496 B]
Get:17 https://nvidia.github.io/nvidia-container-runtime/stable/ubuntu18.04/amd64 Packages [5828 B]
Get:18 https://nvidia.github.io/nvidia-docker/ubuntu18.04/amd64 Packages [4180 B]
Fetched 28.4 kB in 3s (9554 B/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
All packages are up to date.
W: Target Packages (stable/binary-amd64/Packages) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target Packages (stable/binary-all/Packages) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target Translations (stable/i18n/translation-en) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target CNF (stable/cnf/Commands-amd64) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target CNF (stable/cnf/Commands-all) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target Packages (stable/binary-amd64/Packages) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target Packages (stable/binary-all/Packages) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target Translations (stable/i18n/translation-en) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target CNF (stable/cnf/Commands-amd64) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target CNF (stable/cnf/Commands-all) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
kohei@DESKTOP-Q123T6P:~$
```

インストール完了！やっとここまで来た.....

```
kohei@DESKTOP-Q123T6P:~$ sudo apt install -y nvidia-docker2
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libnvidia-container-tools libnvidia-container1 nvidia-container-runtime nvidia-container-toolkit
The following NEW packages will be installed:
  libnvidia-container-tools libnvidia-container1 nvidia-container-runtime nvidia-container-toolkit nvidia-docker2
0 upgraded, 5 newly installed, 0 to remove and 0 not upgraded.
Need to get 1471 kB of archives.
After this operation, 4684 kB of additional disk space will be used.
Get:1 https://nvidia.github.io/libnvidia-container/experimental/ubuntu18.04/amd64 libnvidia-container1 1.3.0~rc.1-1 [66.8 kB]
Get:2 https://nvidia.github.io/libnvidia-container/experimental/ubuntu18.04/amd64 libnvidia-container-tools 1.3.0~rc.1-1 [20.4 kB]
Get:3 https://nvidia.github.io/nvidia-container-runtime/experimental/ubuntu18.04/amd64 nvidia-container-toolkit 1.3.0~rc.2-1 [763 kB]
Get:4 https://nvidia.github.io/nvidia-container-runtime/stable/ubuntu18.04/amd64 nvidia-container-runtime 3.3.0-1 [615 kB]
Get:5 https://nvidia.github.io/nvidia-docker/ubuntu18.04/amd64 nvidia-docker2 2.4.0-1 [5792 B]
Fetched 1471 kB in 1s (1024 kB/s)
Selecting previously unselected package libnvidia-container1:amd64.
(Reading database ... 100906 files and directories currently installed.)
Preparing to unpack .../libnvidia-container1_1.3.0~rc.1-1_amd64.deb ...
Unpacking libnvidia-container1:amd64 (1.3.0~rc.1-1) ...
Selecting previously unselected package libnvidia-container-tools.
Preparing to unpack .../libnvidia-container-tools_1.3.0~rc.1-1_amd64.deb ...
Unpacking libnvidia-container-tools (1.3.0~rc.1-1) ...
Selecting previously unselected package nvidia-container-toolkit.
Preparing to unpack .../nvidia-container-toolkit_1.3.0~rc.2-1_amd64.deb ...
Unpacking nvidia-container-toolkit (1.3.0~rc.2-1) ...
Selecting previously unselected package nvidia-container-runtime.
Preparing to unpack .../nvidia-container-runtime_3.3.0-1_amd64.deb ...
Unpacking nvidia-container-runtime (3.3.0-1) ...
Selecting previously unselected package nvidia-docker2.
Preparing to unpack .../nvidia-docker2_2.4.0-1_all.deb ...
Unpacking nvidia-docker2 (2.4.0-1) ...
Setting up libnvidia-container1:amd64 (1.3.0~rc.1-1) ...
Setting up libnvidia-container-tools (1.3.0~rc.1-1) ...
Setting up nvidia-container-toolkit (1.3.0~rc.2-1) ...
Setting up nvidia-container-runtime (3.3.0-1) ...
Setting up nvidia-docker2 (2.4.0-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9) ...
kohei@DESKTOP-Q123T6P:~$
```

4. デフォルトのランタイムを設定した後、Docker デーモンを再起動してインストールが完了する。

```
sudo systemctl restart docker
```

しかし、ここにきてエラー....

```
kohei@DESKTOP-Q123T6P:~$ sudo systemctl restart docker
System has not been booted with systemd as init system (PID 1). Can't operate.
Failed to connect to bus: Host is down
```

1.6. PID 1 を **systemd** に変更する

インストールは上手くいったそうなので、別の理由かと思いつつ、調べてみた。

参考：【WSL2】systemctl が動かない問題をきちんと解決する

```
System has not been booted with systemd as init system (PID 1). Can't operate.
Failed to connect to bus: Host is down
```

機械翻訳：

システムが systemd で init システム (PID 1) として起動されていません。操作できません。バスに接続できませんでした：ホストがダウンしています。

これは、systemd が PID 1 (init system) ではないので操作できないよ。みたいな感じ。

参考サイトによると

Linux が起動するとき、すべてのプロセスの最初として「systemd」が起動し、すべてのプロセスの親としてふるまう。

実際に以下のコマンドで、WSL2 上のプロセスを確認してみると、PID 1 が /init になっており systemd ではないことが確認できる。

```
ps aux
```

```
kohei@DESKTOP-Q123T6P:~$ ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0      892   572 ?        Ssl   10:40   0:00 /init
root        49  0.0  0.0      900    88 ?        S     10:41   0:00 /init
root       1174  0.6  0.7 1895516 101856 ?        Ssl   15:35   0:25 /usr/bin/dockerd -p /var/run/docker.pid
root       1193  0.5  0.4 1704304 55508 ?        Ssl   15:35   0:22 containerd --config /var/run/docker/containerd/containerd.toml --log-level info
root       2516  0.0  0.0      900    80 ?        Ss    15:57   0:00 /init
root       2517  0.0  0.0      900    88 ?        R     15:57   0:00 /init
kohei     2518  0.0  0.0    10056   5148 pts/1    Ss    15:57   0:00 -bash
kohei     4703  0.0  0.0    10612   3376 pts/1    R+    16:41   0:00 ps aux
kohei@DESKTOP-Q123T6P:~$
```

これが、原因で systemd が動作していない模様...

そこで、以下のリポジトリを用いてこの問題を解決する。

<https://github.com/arkane-systems/genie>

※ この方法は、WSL2 でのみ適用可能。

1. パッケージを追加する。

```
curl -s https://packagecloud.io/install/repositories/arkane-systems/wsl-translinux/script.deb.sh | sudo bash
```

```
kohei@DESKTOP-Q123T6P:~$ curl -s https://packagecloud.io/install/repositories/arkane-systems/wsl-translinux/script.deb.sh | sudo bash
[sudo] password for kohei:
Detected operating system as Ubuntu/focal.
Checking for curl...
Detected curl...
Checking for gpg...
Detected gpg...
Running apt-get update... done.
Installing apt-transport-https... done.
Installing /etc/apt/sources.list.d/arkane-systems_wsl-translinux.list...done.
Importing packagecloud gpg key... done.
Running apt-get update... done.
The repository is setup! You can now install packages.
```

2. 以下のコマンドでインストールする。

```
sudo apt install -y systemd-genie
```

すると、次のようなエラーが表示される。

```
kohei@DESKTOP-Q123T6P:~$ sudo apt install -y systemd-genie
Reading package lists... Done
Building dependency tree
Reading state information... Done
Some packages could not be installed. This may mean that you have
requested an impossible situation or if you are using the unstable
distribution that some required packages have not yet been created
or been moved out of Incoming.
The following information may help to resolve the situation:

The following packages have unmet dependencies:
systemd-genie : Depends: dotnet-runtime-3.1 but it is not installable
E: Unable to correct problems, you have held broken packages.
kohei@DESKTOP-Q123T6P:~$
```

```
The following packages have unmet dependencies:
systemd-genie : Depends: dotnet-runtime-3.1 but it is not installable

E: Unable to correct problems, you have held broken packages.
```

機械翻訳：

次のパッケージには、満たされていない依存関係があります。

systemd-genie : 依存 : dotnet-runtime-3.1 ですが、インストールできません。

E : 問題を修正できません。パッケージが壊れています。

そこで、依存関係の **dotnet-runtime-3.1** をインストールする。

```
sudo apt install dotnet-runtime-3.1
```

しかし、次のようなエラーが表示される。

```
kohei@DESKTOP-Q123T6P:~$ sudo apt install dotnet-runtime-3.1
Reading package lists... Done
Building dependency tree
Reading state information... Done
Package dotnet-runtime-3.1 is not available, but is referred to by another package.
This may mean that the package is missing, has been obsoleted, or
is only available from another source

E: Package 'dotnet-runtime-3.1' has no installation candidate
kohei@DESKTOP-Q123T6P:~$
```

```
Package dotnet-runtime-3.1 is not available, but is referred to by another
package.
This may mean that the package is missing, has been obsoleted, or is only
available from another source

E: Package 'dotnet-runtime-3.1' has no installation candidate
```

機械翻訳：

パッケージ dotnet-runtime-3.1 は使用できませんが、別のパッケージによって参照されています。これは、パッケージが見つからないか、廃止されているか、別のソースからしか入手できないことを意味します。

E：パッケージ 'dotnet-runtime-3.1'にはインストール候補がありません。

なんで、みんな落とせてるのよ....

Microsoft の [Ubuntu に .NET Core SDK または .NET Core ランタイムをインストールする](#) を参考に [dotnet-runtime-3.1](#) をインストールする。

- .Net をインストールする前に、次のコマンドを実行して、信頼された key の一覧に Microsoft パッケージ署名 key を追加し、パッケージリポジトリを追加する。

```
wget https://packages.microsoft.com/config/ubuntu/20.04/packages-microsoft-
prod.deb -O packages-microsoft-prod.deb

sudo dpkg -i packages-microsoft-prod.deb
```

- [.Net Core](#) ランタイムを使用すると、ランタイムを含まない [.Net Core](#) を使用して作成されたアプリを実行することができる。次のコマンドを実行すると、[.Net Core](#) の最も互換性の高いランタイムである [ASP .Net Core](#) ランタイムをインストールできる。

```
sudo apt update; \
sudo apt install -y apt-transport-https && \
sudo apt update && \
sudo apt install -y aspnetcore-runtime-3.1
```

これは、目的の [.Net Core runtime-3.1](#) ではないが、動作するか確認したかったので、一度インストールを試みた。

インストールは成功。

```

Selecting previously unselected package dotnet-host.
(Reading database ... 100944 files and directories currently installed.)
Preparing to unpack .../dotnet-host_3.1.8-1_amd64.deb ...
Unpacking dotnet-host (3.1.8-1) ...
Selecting previously unselected package dotnet-hostfxr-3.1.
Preparing to unpack .../dotnet-hostfxr-3.1_3.1.8-1_amd64.deb ...
Unpacking dotnet-hostfxr-3.1 (3.1.8-1) ...
Selecting previously unselected package dotnet-runtime-deps-3.1.
Preparing to unpack .../dotnet-runtime-deps-3.1_3.1.8-1_amd64.deb ...
Unpacking dotnet-runtime-deps-3.1 (3.1.8-1) ...
Selecting previously unselected package dotnet-runtime-3.1.
Preparing to unpack .../dotnet-runtime-3.1_3.1.8-1_amd64.deb ...
Unpacking dotnet-runtime-3.1 (3.1.8-1) ...
Selecting previously unselected package aspnetcore-runtime-3.1.
Preparing to unpack .../aspnetcore-runtime-3.1_3.1.8-1_amd64.deb ...
Unpacking aspnetcore-runtime-3.1 (3.1.8-1) ...
Setting up dotnet-host (3.1.8-1) ...
Setting up dotnet-runtime-deps-3.1 (3.1.8-1) ...
Setting up dotnet-hostfxr-3.1 (3.1.8-1) ...
Setting up dotnet-runtime-3.1 (3.1.8-1) ...
Setting up aspnetcore-runtime-3.1 (3.1.8-1) ...
Processing triggers for man-db (2.9.1-1) ...
kohei@DESKTOP-Q123T6P:~$

```

先ほど失敗した、**genie** のインストールも試してみる。

```
sudo apt install -y systemd-genie
```

結果、インストールに成功した。

```

kohei@DESKTOP-Q123T6P:~$ sudo apt install -y systemd-genie
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  daemonize
The following NEW packages will be installed:
  daemonize systemd-genie
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 208 kB of archives.
After this operation, 705 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu focal/universe amd64 daemonize amd64 1.7.8-1 [11.9 kB]
Get:2 https://packagecloud.io/arkane-systems/wsl-translinux/ubuntu focal/main amd64 systemd-genie amd64 1.28 [197 kB]
Fetched 208 kB in 1s (205 kB/s)
Selecting previously unselected package daemonize.
(Reading database ... 101296 files and directories currently installed.)
Preparing to unpack .../daemonize_1.7.8-1_amd64.deb ...
Unpacking daemonize (1.7.8-1) ...
Selecting previously unselected package systemd-genie.
Preparing to unpack .../systemd-genie_1.28_amd64.deb ...
Unpacking systemd-genie (1.28) ...
Setting up daemonize (1.7.8-1) ...
Setting up systemd-genie (1.28) ...
Processing triggers for man-db (2.9.1-1) ...
kohei@DESKTOP-Q123T6P:~$

```

3. インストールした **genie** を使って PID 1 を変更する。

genie をインストールしただけでは、PID に変化はなかった。

```

kohei@DESKTOP-Q123T6P:~$ ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0     892   572 ?        Ssl   10:40    0:00 /init
root        49  0.0  0.0     900    88 ?        S    10:41    0:00 /init
root       1174  0.5  0.7 1895772 101856 ?        Sl    15:35    0:48 /usr/bin/dockerd -p /var/run/docker.pid
root       1193  0.5  0.4 1704560 55508 ?        Ssl   15:35    0:43 containerd --config /var/run/docker/containerd/conta
root       2516  0.0  0.0     900    80 ?        Ss   15:57    0:00 /init
root       2517  0.0  0.0     900    88 ?        R    15:57    0:00 /init
kohei      2518  0.0  0.0   10056   5148 pts/1    Ss   15:57    0:00 -bash
kohei      8616  0.0  0.0   10612   3364 pts/1    R+   17:49    0:00 ps aux
kohei@DESKTOP-Q123T6P:~$

```

そのために、以下のコマンドを実行する。

```
genie -s
```

このコマンドを実行すると、PID 1 がしっかりと **systemd** に変化した。

```
kohei@DESKTOP-Q123T6P:~$ ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0  17320  1312 ?        Ss   17:52   0:00 systemd
root        44  4.1  0.1  37100 13008 ?        Ss   17:52   0:00 /lib/systemd/systemd-journald
root        61  0.0  0.0  10152   3716 pts/1    Ss   17:52   0:00 runuser -l kohei -w INSIDE_GENIE_WSL_DISTRO_NAME,kohei
root        81  3.6  0.0  19148   5200 ?        Ss   17:52   0:00 /lib/systemd/systemd-udevd
systemd+   84  1.4  0.0  18688   7812 ?        Ss   17:52   0:00 /lib/systemd/systemd-networkd
systemd+  279  1.2  0.0  23524  12528 ?        Ss   17:52   0:00 /lib/systemd/systemd-resolved
systemd+  280  1.2  0.0  90474   6436 ?        Ss   17:52   0:00 /lib/systemd/systemd-timesyncd
root       285  0.7  0.0  241028 10888 ?        Ss   17:52   0:00 /usr/lib/accounts-service/accounts-daemon
avahi      286  0.2  0.0   8616   3692 ?        Ss   17:52   0:00 avahi-daemon: running [DESKTOP-Q123T6P-wsl.local]
message+  287  4.8  0.0   7504   4880 ?        Ss   17:52   0:00 /usr/bin/dbus-daemon --system --address=systemd: --root
root       296  0.0  0.0   81824  3758 ?        Ss   17:52   0:00 /usr/sbin/irqbalance --foreground
root       298  1.4  0.1  28252  18184 ?        Ss   17:52   0:00 /usr/bin/python3 /usr/bin/networkd-dispatcher --runroot
systemd+  304  1.7  0.0  224336  4376 ?        Ss   17:52   0:00 /usr/sbin/plymouthd --no-LOGO
root       308  0.2  0.0  234708  5692 ?        Ss   17:52   0:00 /usr/libexec/switcheroo-control
root       310  4.0  0.0  17020   7992 ?        Ss   17:52   0:00 /lib/systemd/systemd-logind
root       314  0.0  0.0   13676   5500 ?        Ss   17:52   0:00 /sbin/wpa_supplicant -u -s -o /run/wpa_supplicant
avahi      320  0.0  0.0   8332   328 ?        Ss   17:52   0:00 avahi-daemon: chroot helper
root       341  1.4  0.3  1491988 51124 ?       Ss   17:52   0:00 /usr/bin/containers
root       344  1.1  0.0  313628  12244 ?       Ss   17:52   0:00 /usr/sbin/ModemManager --filter-policy=strict
root       348  1.2  0.0   16428   6196 ?        Ss   17:52   0:00 /lib/systemd/systemd-hostnamed
root       378  0.9  0.1  108044  20964 ?       Ss   17:52   0:00 /usr/bin/python3 /usr/share/unattended-upgrades/unroot
daemon    407  0.0  0.0   3796   2428 ?        Ss   17:52   0:00 /usr/sbin/udm -f
root       491  0.5  0.2  642356 29572 ?        Ss   17:52   0:00 /usr/bin/snap wait system seed.loaded
root       960  2.4  0.2  239508  3084 ?        Ss   17:52   0:02 /usr/lib/snapd/snapd
root     1005  2.3  0.0  238624 10550 ?        Ss   17:53   0:00 /usr/sbin/gdm3
root     1019  0.6  0.0   19548   4228 ?        Ss   17:53   0:00 /lib/systemd/systemd-udevd
dm         1021  2.6  0.0   46832  10086 ?        Ss   17:53   0:00 /lib/systemd/systemd-user
root     1022  0.0  0.0  174364   4300 ?        Ss   17:53   0:00 (sd-pam)
rtkit     1064  0.3  0.0   152928  2968 ?        Ss   17:53   0:00 /usr/libexec/rtkit-daemon
dm         1143  0.5  0.0   70896   4072 ?        Ss   17:53   0:00 /usr/bin/dbus-daemon --session --address=systemd:
root     1274  2.4  0.5  1525416 89244 ?       Ss   17:53   0:00 /usr/bin/dockerd -H fd:// --containerd=/run/containers
kohei    1407  0.0  0.0   5892  2920 pts/1    R+   17:53   0:00 ps aux
```

1.7. Docker daemon を再起動

先ほどできなかった、**docker daemon**の再起動を行う。

```
sudo systemctl restart docker
```

結果は、別のエラー....

```
kohei@DESKTOP-Q123T6P-wsl:~$ sudo systemctl restart docker
[sudo] password for kohei:
Job for docker.service failed because the control process exited with error code.
See "systemctl status docker.service" and "journalctl -xe" for details.
```

```
Job for docker.service failed because the control process exited with error code.
See "systemctl status docker.service" and "journalctl -xe" for details.
```

機械翻訳：

制御プロセスがエラーコードで終了したため、docker.service のジョブが失敗しました。

詳細については、「systemctl status docker.service」および「journalctl -xe」を参照してください。

docker のリスタートが行いたかったので、ここで一度再起動してみることに。

しかし、再起動すると PID 1 がまた **init** に戻ることが判明！

```
kohei@DESKTOP-Q123T6P:~$ ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  8.0  0.0   892    552 ?        Sl   18:51   0:00 /init
root         7  0.0  0.0    900     84 ?        Ss   18:51   0:00 /init
root         8  0.0  0.0    900     92 ?        R    18:51   0:00 /init
kohei        9  1.0  0.0  10056  4928 pts/0    Ss   18:51   0:00 -bash
kohei       22  0.0  0.0  10612  3360 pts/0    R+   18:51   0:00 ps aux
```

そのため、WSL2 では起動の度に


```
genie -s
```

コマンドを打たなければならない。

1.8. CUDA コンテナの実行

ベースの CUDA コンテナを実行することで、機能しているセットアップをテストする。

```
sudo docker run --rm --gpus all nvidia/cuda:11.0-base nvidia-smi
```

しかし、次のようなエラーが発生。

```
kohei@DESKTOP-Q123T6P-ws:~$ sudo docker run --rm --gpus all nvidia/cuda:11.0-base nvidia-smi
[sudo] password for kohei:
Unable to find image 'nvidia/cuda:11.0-base' locally
11.0-base: Pulling from nvidia/cuda
54ee1f796a1e: Pull complete
f7bfea53ad12: Pull complete
46d371e02073: Pull complete
b66c17bbf772: Pull complete
3642f1a6dfb3: Pull complete
e5ca55b8b4b9: Pull complete
155bc0332b0a: Pull complete
Digest: sha256:774ca3d612de15213102c2dbbba55df44dc5cf9870ca2be6c6e9c627fa63d67a
Status: Downloaded newer image for nvidia/cuda:11.0-base
docker: Error response from daemon: OCI runtime create failed: container_linux.go:349: starting container process caused "process_linux.go:449: container init caused \"process_linux.go:432: running prestart hook 0 caused \\\"error running hook: exit status 1, stdout: , stderr: nvidia-container-cli: initialization error: nvml error: driver not loaded\\\"n\\\"\\\"\": unknown.
```

```
docker: Error response from daemon: OCI runtime create failed:
container_linux.go:349: starting container process caused "process_linux.go:449:
container init caused \"process_linux.go:432: running prestart hook 0 caused
\\\"error running hook: exit status 1, stdout: , stderr: nvidia-container-cli:
initialization error: nvml error: driver not loaded\\\"n\\\"\\\"\": unknown.
```

機械翻訳：

```
docker : デーモンからのエラー応答 : OCI ランタイムの作成に失敗しました : container_linux.go :
349 : コンテナプロセスの開始により「process_linux.go :
449 : コンテナの初期化により\"process_linux.go :
432 : 実行前開始フック 0 により\"エラーが発生しました。

実行中のフック : 終了ステータス 1、標準出力 : 、標準エラー : nvidia-container-cli : 初期化エラー :
nvml エラー : ドライバーが読み込まれていません\\n\\\" \" : 不明です。
```

ドライバーが読み込まれていませんという文があることから、**nvidia-driver** が必要なのではないかと考えた。

そこで、以下のコマンドを実行する。

```
nvidia-sim
```

すると、次のようなメッセージが表示された。

```
NVIDIA-SMI has failed because it couldn't communicate with the NVIDIA driver. Make
sure that the latest NVIDIA driver is installed and running.
```

機械翻訳：

NVIDIA-SMI は、NVIDIA ドライバーと通信できなかったために失敗しました。最新の NVIDIA ドライバーがインストールされ、実行されていることを確認します。

1.9. NVIDIA Driver のインストール

参考

- [ubuntu18.04 に nvidia ドライバを入れるの苦労した](#)
- [Ubuntu 20.04 セットアップ](#)

NVIDIA Driver のインストールには、いくつか方法がある。

1. `ubuntu-drivers` コマンドでインストールする
2. `software & Updates` の `Additional Drivers` からインストールする
3. NVIDIA 公式からドライバをダウンロードしてインストールする。

ここでは、1. の方法でインストールする。

(Optional) PPA を追加する

```
sudo add-apt-repository ppa:graphics-drivers/ppa

sudo apt update
```

途中で一度 `Enter` を押す場所がある。

```
kohei@DESKTOP-Q123T6P-ws1:~$ sudo add-apt-repository ppa:graphics-drivers/ppa
[sudo] password for kohei:
Sorry, try again.
[sudo] password for kohei:
Fresh drivers from upstream, currently shipping Nvidia.
```

```

kohei@DESKTOP-Q123T6P-ws1:~$ sudo apt update
Get:1 file:/var/cuda-repo-ubuntu2004-11-0-local InRelease
Ign:1 file:/var/cuda-repo-ubuntu2004-11-0-local InRelease
Get:2 file:/var/cuda-repo-ubuntu2004-11-0-local Release [564 B]
Get:2 file:/var/cuda-repo-ubuntu2004-11-0-local Release [564 B]
Hit:5 http://archive.ubuntu.com/ubuntu focal InRelease
Hit:6 https://download.docker.com/linux/ubuntu focal InRelease
Hit:7 https://nvidia.github.io/libnvidia-container/experimental/ubuntu18.04/amd64 InRelease
Hit:8 https://nvidia.github.io/libnvidia-container-runtime/experimental/ubuntu18.04/amd64 InRelease
Hit:9 https://nvidia.github.io/libnvidia-container/stable/ubuntu18.04/amd64 InRelease
Hit:10 https://nvidia.github.io/nvidia-container-runtime/stable/ubuntu18.04/amd64 InRelease
Hit:11 https://nvidia.github.io/nvidia-docker/ubuntu18.04/amd64 InRelease
Hit:12 http://archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:13 http://security.ubuntu.com/ubuntu focal-security InRelease
Hit:14 http://archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:15 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu focal InRelease
Hit:4 https://packagecloud.io/arkane-systems/wsl-translinux/ubuntu focal InRelease
Err:16 https://packages.microsoft.com/ubuntu/20.04/prod focal InRelease
Temporary failure resolving 'packages.microsoft.com'
Reading package lists... Done
Building dependency tree
Reading state information... Done
All packages are up to date.
W: Target Packages (stable/binary-amd64/Packages) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target Packages (stable/binary-all/Packages) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target Translations (stable/i18n/Translation-en) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target CNF (stable/cnf/Commands-amd64) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target CNF (stable/cnf/Commands-all) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Failed to fetch https://packages.microsoft.com/ubuntu/20.04/prod/dists/focal/InRelease Temporary failure resolving 'packages.microsoft.com'
W: Some index files failed to download. They have been ignored, or old ones used instead.
W: Target Packages (stable/binary-amd64/Packages) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target Packages (stable/binary-all/Packages) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target Translations (stable/i18n/Translation-en) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target CNF (stable/cnf/Commands-amd64) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
W: Target CNF (stable/cnf/Commands-all) is configured multiple times in /etc/apt/sources.list:50 and /etc/apt/sources.list.d/docker.list:1
kohei@DESKTOP-Q123T6P-ws1:~$

```

(Optional) 推奨ドライバを確認する。

推奨ドライバを自動的に確認するために、依存パッケージを先にインストールする。

```
sudo apt install -y ubuntu-drivers-common
```

推奨ドライバを確認する。

```
ubuntu-drivers devices
```

このコマンドを実行することで、推奨ドライバを確認することができるはずだが、自分のパソコンでは表示されなかった。

```

kohei@DESKTOP-Q123T6P-ws1:~$ ubuntu-drivers devices
kohei@DESKTOP-Q123T6P-ws1:~$

```

しかし、今回はこれを無視して次の手順に進む。

ドライバをインストールする

自動で推奨ドライバをインストールする。

```
sudo ubuntu-drivers autoinstall
```

結果は、ドライバを発見することができなかった。

```

kohai@DESKTOP-Q123T6P-ws1:~$ sudo ubuntu-drivers autoinstall
No drivers found for installation.

```

既存のドライバが入ってるから...?

既存ドライバのアンインストール

以前インストールしていたドライバ・CUDA をアンインストールする。

```
sudo apt --purge remove nvidia-*  
  
sudo apt-get --purge remove cuda-*
```

どうも、nvidia driver はインストールされていなかったみたい....

```
kohei@DESKTOP-Q123T6P-ws1:~$ sudo apt --purge remove nvidia-*  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
E: Unable to locate package nvidia-*  
kohei@DESKTOP-Q123T6P-ws1:~$ sudo apt-get --purge remove cuda-*  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
E: Unable to locate package cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb  
E: Couldn't find any package by glob 'cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb'  
E: Couldn't find any package by regex 'cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb'  
kohei@DESKTOP-Q123T6P-ws1:~$
```

全てアンインストールしたが、`autoinstall` できなかった。

```
kohei@DESKTOP-Q123T6P-ws1:~$ sudo ubuntu-drivers autoinstall  
No drivers found for installation.
```

NVIDIA の公式から **Driver** をインストールする

NVIDIA 公式 に自分の製品情報を入力する。

NVIDIAドライバダウンロード

オプション1: エヌビディア製品用ドライバを手動検索する

製品のタイプ: GeForce

製品シリーズ: GeForce GTX 16 Series (Notebooks)

製品ファミリー: GeForce GTX 1660 Ti

オペレーティングシステム: Linux 64-bit

ダウンロードタイプ: Linux Long Lived

言語: Japanese

検索

そして、「検索ボタン」を押すと、自分の製品に最適な Driver の version が表示される。



このバージョンを参考に、Driver をインストールする。

```
sudo add-apt-repository ppa:graphics-drivers/ppa  
  
sudo apt update  
  
sudo apt install nvidia-driver-430
```

インストールすることができた。

```
kohei@DESKTOP-Q123T6P-ws1:~$ sudo apt install nvidia-driver-430  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following additional packages will be installed:  
  nvidia-driver-440  
The following NEW packages will be installed:  
  nvidia-driver-430 nvidia-driver-440  
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.  
Need to get 12.9 kB of archives.  
After this operation, 38.9 kB of additional disk space will be used.  
Do you want to continue? [Y/n] y  
Get:1 http://archive.ubuntu.com/ubuntu focal-updates/restricted amd64 nvidia-driver-430 amd64 440.100-0ubuntu0.20.04.1 [7340 B]  
Get:2 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu focal/main amd64 nvidia-driver-440 amd64 450.66-0ubuntu0.20.04.1 [5516 B]  
Fetched 12.9 kB in 1s (23.8 kB/s)  
Selecting previously unselected package nvidia-driver-440.  
(Reading database ... 101468 files and directories currently installed.)  
Preparing to unpack .../nvidia-driver-440_450.66-0ubuntu0.20.04.1_amd64.deb ...  
Unpacking nvidia-driver-440 (450.66-0ubuntu0.20.04.1) ...  
Selecting previously unselected package nvidia-driver-430.  
Preparing to unpack .../nvidia-driver-430_440.100-0ubuntu0.20.04.1_amd64.deb ...  
Unpacking nvidia-driver-430 (440.100-0ubuntu0.20.04.1) ...  
Setting up nvidia-driver-440 (450.66-0ubuntu0.20.04.1) ...  
Setting up nvidia-driver-430 (440.100-0ubuntu0.20.04.1) ...  
kohei@DESKTOP-Q123T6P-ws1:~$
```

Driver をインストールしても、エラーは消えなかった。

```
kohei@DESKTOP-Q123T6P-ws1:~$ nvidia-smi  
NVIDIA-SMI has failed because it couldn't communicate with the NVIDIA driver. Make sure that the latest NVIDIA driver is installed and running.
```

CUDA Toolkit を再インストールする

CUDA がインストールされていないかを確認する。

```
nvcc -V
```

```
kohei@DESKTOP-Q123T6P-ws1:~$ nvcc -V  
Command 'nvcc' not found, but can be installed with:  
sudo apt install nvidia-cuda-toolkit  
kohei@DESKTOP-Q123T6P-ws1:~$
```

メッセージに書かれている通りに、インストールする

```
sudo apt install nvidia-cuda-toolkit
```

再インストール完了。

```
kohei@DESKTOP-Q123T6P-ws1:~$ sudo apt install nvidia-cuda-toolkit
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
```

先ほどのコマンドを再度実行し、**CUDA-Toolkit** がインストールできたかを確認。

```
nvcc -V
```

version が表示されたことから、インストールを確認。

```
kohei@DESKTOP-Q123T6P-ws1:~$ nvcc -V
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2019 NVIDIA Corporation
Built on Sun_Jul_28_19:07:16_PDT_2019
Cuda compilation tools, release 10.1, V10.1.243
kohei@DESKTOP-Q123T6P-ws1:~$
```

しかし、やはり動作しない。

```
kohei@DESKTOP-Q123T6P-ws1:~$ nvidia-smi
NVIDIA-SMI has failed because it couldn't communicate with the NVIDIA driver. Make sure that the latest NVIDIA driver is installed and running.
```

仕方がないので今日はここまで...