

주의

Tensorflow/PyTorch GPU 가속은 공식적으로 NVIDIA 만 지원됩니다.

AMD GPU 를 사용중인 경우 ROCm 을 사용가능합니다 (Only LINUX)

Intel GPU 를 사용중인 경우 oneAPI 나 direct-ML, ONNX 등을 고려해 볼 수 있습니다.

(Intel/AMD 는 Direct-ML 이나 ONNX 등을 사용할 수 있으나 권장되지 않습니다)

이 가이드에서는 NVIDIA(+CPU)만 설명하므로

다른 방법으로 생기는 문제에 대해서는 보증하지 않으며

문제가 발생하는 경우 해당 포럼을 방문하여 문제 해결 바랍니다.

준비사항 GPU

NVIDIA GPU with CUDA support and Driver support.

Windows 10 v2004 or windows 11

<https://learn.microsoft.com/en-GB/cpp/windows/latest-supported-vc-redist?view=msvc-170>

MSVC 2015,2017,2019

[GPU 설치로 >>](#)

[GPU-Windows Native](#)

[CPU 설치로 >>](#)

[VSCODE 에서 확인하기 >>](#)

[P.S.](#)

GPU 설치

1. Driver 설치

<https://developer.nvidia.com/cuda-toolkit>

위 페이지에서 Cuda Toolkit 을 설치합니다.

Download Now -> Select your environment(Windows in this case) -> Choose the Installer Type

Operating System	Linux	Windows		
Architecture	x86_64			
Version	10	11	Server 2019	Server 2022
Installer Type	exe (local)	exe (network)		

Local 은 package 전체, network 는 다운로더만 받아옵니다.

특정 버전으로 맞추고 싶은 경우 Download Page 하단 Resources -> Archive of Pre...로 이동하여 버전을 선택 후 받는다.

Resources

- [CUDA Documentation/Release Notes](#)
- [MacOS Tools](#)
- [Training](#)
- [Sample Code](#)
- [Forums](#)
- [Archive of Previous CUDA Releases](#)
- [FAQ](#)
- [Open Source Packages](#)
- [Submit a Bug](#)
- [Tarball and Zip Archive Deliverables](#)

CUDA Toolkit Archive

[Home](#)

Previous releases of the CUDA Toolkit, GPU Computing SDK, documentation and developer drivers can be found using the links below. Please select the release you want from the list below, and be sure to check www.nvidia.com/drivers for more recent production drivers appropriate for your hardware configuration.

[Download Latest CUDA Toolkit](#)

[Learn More about CUDA Toolkit](#)

Latest Release


[CUDA Toolkit 12.2.2 \(August 2023\)](#), [Versioned Online Documentation](#)

Archived Releases

[CUDA Toolkit 12.2.1 \(July 2023\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 12.2.0 \(June 2023\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 12.1.1 \(April 2023\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 12.1.0 \(February 2023\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 12.0.1 \(January 2023\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 12.0.0 \(December 2022\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 11.8.0 \(October 2022\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 11.7.1 \(August 2022\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 11.7.0 \(May 2022\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 11.6.2 \(March 2022\)](#), [Versioned Online Documentation](#)

Download Installer for Windows 11 x86_64

The base installer is available for download below.

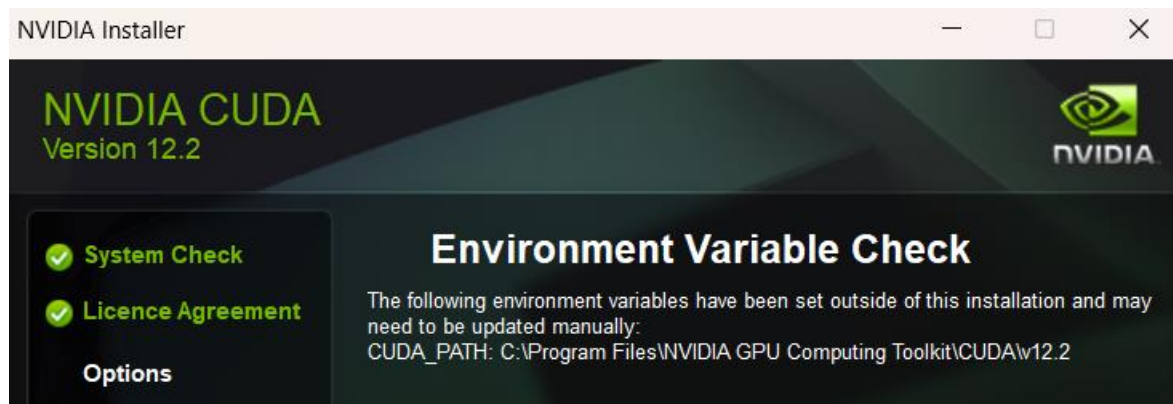
Base Installer Download (3.0 GB) 

Installation Instructions:

1. Double click cuda_12.2.2_537.13_windows.exe
2. Follow on-screen prompts

The checksums for the installer and patches can be found in [Installer Checksums](#).
For further information, see the [Installation Guide for Microsoft Windows](#) and the [CUDA Quick Start Guide](#).

Download 를 눌러 Installer 를 받고 설치한다, 통상 Driver 처럼 진행하면 된다.



환경변수 설정을 별도로 요구하는 경우 설치 완료 후 해당 환경변수를 추가/수정 한다.

2. cuDNN 설치

<https://developer.nvidia.com/cudnn>

위 링크에서 Download cuDNN 으로 이동한다.

NVIDIA 계정이 필수이므로 가입/로그인 후 진행한다.

cuDNN Download

NVIDIA cuDNN is a GPU-accelerated library of primitives for deep neural networks.

☒ **I Agree To the Terms of the cuDNN Software License Agreement**

Note: Please refer to the Installation Guide for release prerequisites, including supp

For more information, refer to the cuDNN Developer Guide, Installation Guide and R

Download cuDNN v8.9.4 (August 8th, 2023), for CUDA 12.x

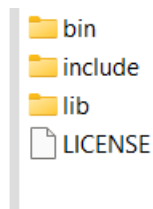
Download cuDNN v8.9.4 (August 8th, 2023), for CUDA 11.x

Archived cuDNN Releases

약관 동의 후 설치된 CUDA 버전과 환경에 맞게 받는다. (구버전 필요 시 Archived 로 이동)

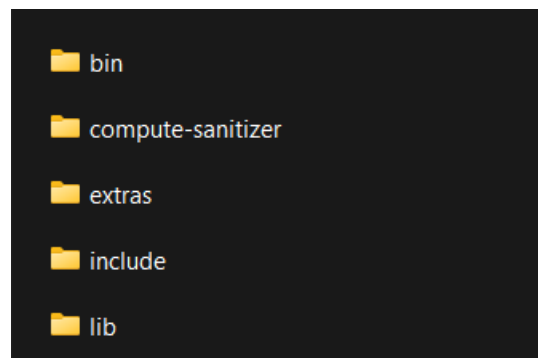
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA 로 이동하여(기타 경로 미 지정 시)

자신이 설치한 CUDA 버전으로 이동한 다음



cuDNN 압축폴더 내 폴더 3 개를

CUDA 설치경로에 붙여넣는다(덮어쓰기)



```
PS C:\Users\ [redacted] > nvidia-smi
Wed Sep  6 00:16:03 2023

+-----+
| NVIDIA-SMI 537.13              | Driver Version: 537.13      | CUDA Version: 12.2          |
+-----+-----+-----+-----+-----+-----+
| GPU  Name    Perf          TCC/WDDM  | Bus-Id      Disp.A | Volatile Uncorr. ECC |
| Fan  Temp    Perf          Pwr:Usage/Cap|           Memory-Usage | GPU-Util  Compute M. |
|=====+=====+=====+=====+=====+=====+
|   0   NVIDIA GeForce RTX 3090   WDDM  | 00000000:0B:00.0 On  |         31%         N/A |
| 49%   49C    P5              50W / 420W | 1779MiB / 24576MiB |         Default      N/A |
+-----+-----+-----+-----+-----+-----+

Processes:
+-----+-----+-----+-----+-----+-----+
| GPU  GI  CI           PID  Type  Process name                        | GPU Memory |
|   ID  ID  ID                                     |      Usage |
+-----+-----+-----+-----+-----+-----+

```

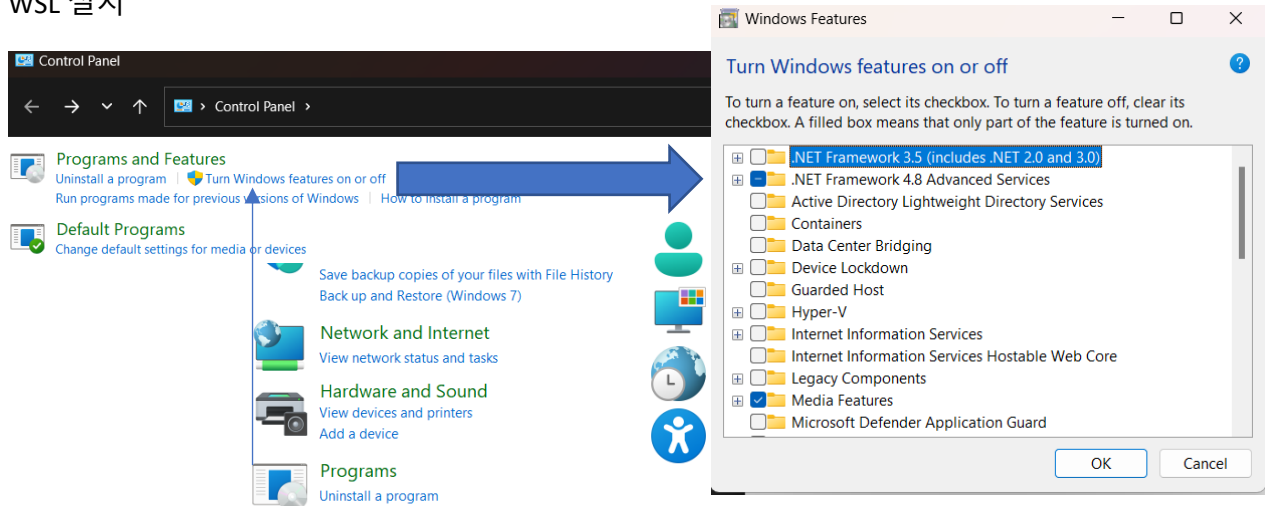
재부팅후 nvidia-smi 로 gpu 드라이버 상태 확인 후

```
PS C:\Users\ [redacted] > nvcc --version
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2023 NVIDIA Corporation
Built on Tue_Aug_15_22:09:35_Pacific_Daylight_Time_2023
Cuda compilation tools, release 12.2, V12.2.140
Build cuda_12.2.r12.2/compiler.33191640_0

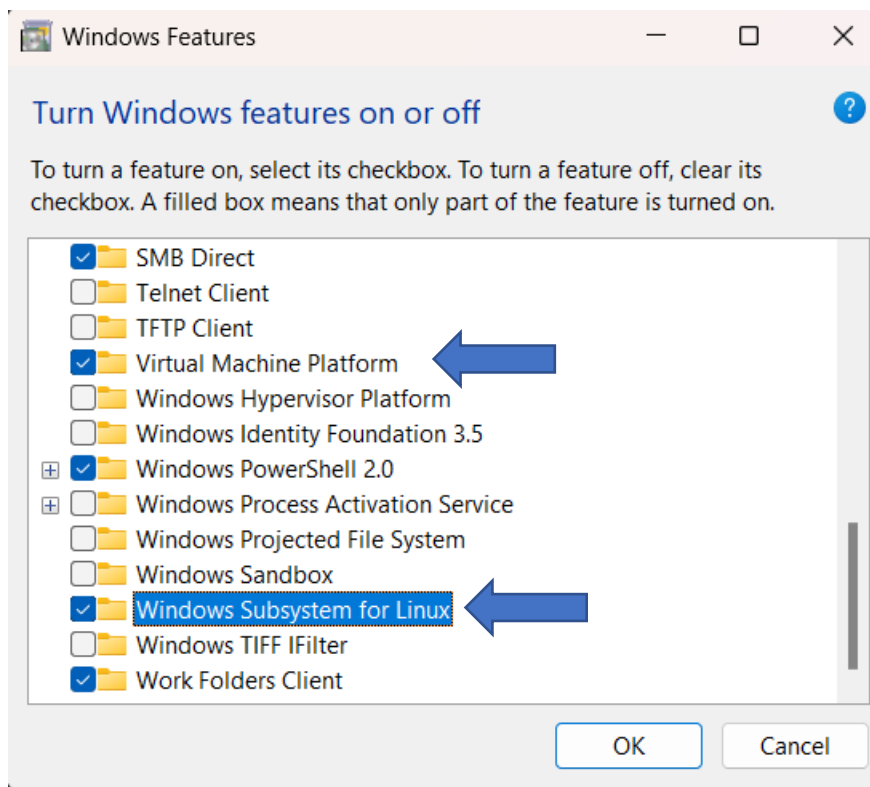
```

Nvcc --version 으로 cuda 버전을 확인한다.

3. WSL 설치



Control -> Programs -> Programs and Features -> Turn Windows features on or off 를 선택한다



위 화면에서 VM Platform 과 WSL 이 있는지 확인한다.

WSL 과 VM Platform 이 보이지 않거나 설치중 오류 발생 시

CPU 로 진행한다.

Terminal 에서(Powershell Recommended)

Wsl 이 설치되어 있지 않은 경우

wsl --install -> Enter 후 재부팅 한다.

WSL 이 이미 설치되어 있거나 설치완료된 경우 (WSL2 Recommended -> Version 2)

wsl --update 실행하여 update 한다.

```
PS C:\User\ [redacted] > wsl -l -v
```

NAME	STATE	VERSION
* docker-desktop	Stopped	2
docker-desktop-data	Stopped	2

wsl -l -v 현재 설치된 배포판을 확인한다. (최초 설치 시 아무것도 표시되지 않는다.)

wsl -l -o 현재 설치 가능한 배포판 목록을 온라인에서 조회

```
PS C:\Users\ [redacted] > wsl -l -o
```

The following is a list of valid distributions that can be installed.
Install using 'wsl.exe --install <Distro>'.

NAME	FRIENDLY NAME
Ubuntu	Ubuntu
Debian	Debian GNU/Linux
kali-linux	Kali Linux Rolling
Ubuntu-18.04	Ubuntu 18.04 LTS
Ubuntu-20.04	Ubuntu 20.04 LTS
Ubuntu-22.04	Ubuntu 22.04 LTS
OracleLinux_7_9	Oracle Linux 7.9
OracleLinux_8_7	Oracle Linux 8.7
OracleLinux_9_1	Oracle Linux 9.1
openSUSE-Leap-15.5	openSUSE Leap 15.5
SUSE-Linux-Enterprise-Server-15-SP4	SUSE Linux Enterprise Server 15 SP4
SUSE-Linux-Enterprise-15-SP5	SUSE Linux Enterprise 15 SP5
openSUSE-Tumbleweed	openSUSE Tumbleweed

위 목록에서 Ubuntu-22.04 를 설치한다. (다른 버전 필요 시 취사선택)

wsl --install -d Ubuntu-22.04

Terminal or 별도로 Ubuntu-22.04 검색 -> 실행 후

sudo apt-get update && sudo apt-get upgrade -y 하여 업데이트를 진행한다.

Linux Terminal 에서

sudo apt-key del 7fa2af80

```
~$ sudo apt-key del 7fa2af80
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
```

기존 GPG 키를 제거한다.

WSL 용 CUDA Toolkit 설치(Recommended)

wget https://developer.download.nvidia.com/compute/cuda/repos/wsl-ubuntu/x86_64/cuda-wsl-ubuntu.pin

```
~$ wget https://developer.download.nvidia.com/compute/cuda/repos/wsl-ubuntu/x86_64/cuda-wsl-ubuntu.pin
--2023-09-06 00:44:50-- https://developer.download.nvidia.com/compute/cuda/repos/wsl-ubuntu/x86_64/cuda-wsl-ubuntu.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-wsl-ubuntu.pin'

cuda-wsl-ubuntu.pin
=====>]          190 --.-KB/s   in 0s
2023-09-06 00:44:51 (21.4 MB/s) - 'cuda-wsl-ubuntu.pin' saved [190/190]
```

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

wget https://developer.download.nvidia.com/compute/cuda/12.2.2/local_installers/cuda-repo-wsl-ubuntu-12-2-local_12.2.2-1_amd64.deb

```
~$ sudo mv cuda-wsl-ubuntu.pin /etc/apt/preferences.d/cuda-repository-pin-600
~$ wget https://developer.download.nvidia.com/compute/cuda/12.2.2/local_installers/cuda-repo-wsl-ubuntu-12-2-local_12.2.2-1_amd64.deb
--2023-09-06 00:46:37-- https://developer.download.nvidia.com/compute/cuda/12.2.2/local_installers/cuda-repo-wsl-ubuntu-12-2-local_12.2.2-1_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: 2875296226 (2.7G) [application/x-deb]
Saving to: 'cuda-repo-wsl-ubuntu-12-2-local_12.2.2-1_amd64.deb'

cuda-repo-wsl-ubuntu-12-2-local_12.2.2-1_amd64.deb
=====>]          2.68G  42.0MB/s   in 58s
2023-09-06 00:47:36 (47.0 MB/s) - 'cuda-repo-wsl-ubuntu-12-2-local_12.2.2-1_amd64.deb' saved [2875296226/2875296226]
```

sudo dpkg -i cuda-repo-wsl-ubuntu-12-2-local_12.2.2-1_amd64.deb

```
~$ sudo dpkg -i cuda-repo-wsl-ubuntu-12-2-local_12.2.2-1_amd64.deb
Selecting previously unselected package cuda-repo-wsl-ubuntu-12-2-local.
(Reading database ... 24157 files and directories currently installed.)
Preparing to unpack cuda-repo-wsl-ubuntu-12-2-local_12.2.2-1_amd64.deb ...
Unpacking cuda-repo-wsl-ubuntu-12-2-local (12.2.2-1) ...
Setting up cuda-repo-wsl-ubuntu-12-2-local (12.2.2-1) ...

The public cuda-repo-wsl-ubuntu-12-2-local GPG key does not appear to be installed.
To install the key, run this command:
sudo cp /var/cuda-repo-wsl-ubuntu-12-2-local/cuda-48257546-keyring.gpg /usr/share/keyrings/
```

```
sudo cp /var/cuda-repo-wsl-ubuntu-12-2-local/cuda-*-keyring.gpg /usr/share/keyrings/
```

```
sudo apt-get update
```

```
$ sudo cp /var/cuda-repo-wsl-ubuntu-12-2-local/cuda-*-keyring.gpg /usr/share/keyrings/
$ sudo apt-get update
Get:1 file:/var/cuda-repo-wsl-ubuntu-12-2-local InRelease [1572 B]
Get:1 file:/var/cuda-repo-wsl-ubuntu-12-2-local InRelease [1572 B]
Get:2 file:/var/cuda-repo-wsl-ubuntu-12-2-local Packages [18.3 kB]
Hit:3 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:4 http://archive.ubuntu.com/ubuntu jammy InRelease
Hit:5 http://archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:6 http://archive.ubuntu.com/ubuntu jammy-backports InRelease
Reading package lists... Done
```

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

```
Adding debian:COMODO_RSA_Certification_Authority.pem
Adding debian:QuoVadis_Root_CA_2_G3.pem
Adding debian:AffirmTrust_Premium.pem
Adding debian:GlobalSign_ECC_Root_CA_-_R5.pem
Adding debian:Entrust.net_Premium_2048_Secure_Server_CA.pem
Adding debian:Buypass_Class_3_Root_CA.pem
done.
Setting up default-jre-headless (2:1.11-72build2) ...
Setting up default-jre (2:1.11-72build2) ...
Setting up cuda-nsight-12-2 (12.2.144-1) ...
Setting up cuda-nvvp-12-2 (12.2.142-1) ...
Setting up cuda-visual-tools-12-2 (12.2.2-1) ...
Setting up cuda-tools-12-2 (12.2.2-1) ...
Setting up cuda-toolkit-12-2 (12.2.2-1) ...
Setting up cuda-12-2 (12.2.2-1) ...
Setting up cuda (12.2.2-1) ...
Processing triggers for libgl1-0:amd64 (2.72.4-0ubuntu2.2) ...
Processing triggers for libc-bin (2.35-0ubuntu3.1) ...
/sbin/ldconfig.real: /usr/lib/wsl/lib/libcuda.so.1 is not a symbolic link

Processing triggers for man-db (2.10.2-1) ...
Processing triggers for ca-certificates (20230311ubuntu0.22.04.1) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
done.
Setting up at-spi2-core (2.44.0-3) ...
```

위 커맨드를 한줄씩 실행한다.

이후 `nvidia-smi` 와 `nvcc --version` 으로 확인한다.

4. Tensorflow-GPU 설치

Linux Terminal 에서 진행한다.

curl https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh -o Miniconda3-latest-Linux-x86_64.sh

```
~$ curl https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh -o Miniconda3-latest-Linux-x86_64.sh
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total     Spent    Left     Speed
100 98.4M  100 98.4M    0     0  67.3M      0  0:00:01  0:00:01 --:--:-- 67.4M
```

bash Miniconda3-latest-Linux-x86_64.sh

```
~$ bash Miniconda3-latest-Linux-x86_64.sh

Welcome to Miniconda3 py311_23.5.2-0

In order to continue the installation process, please review the license
agreement.
Please, press ENTER to continue
>>>

Miniconda3 will now be installed into this location:
/home/USER/miniconda3

- Press ENTER to confirm the location
- Press CTRL-C to abort the installation
- Or specify a different location below

[/home/USER/miniconda3] >>> _
```

별도 경로 지정을 원하는 경우 입력한다. (기본 /home/USER/miniconda3)

```
Preparing transaction: done
Executing transaction: done
installation finished.
Do you wish the installer to initialize Miniconda3
by running conda init? [yes|no]
[no] >>>

==> For changes to take effect, close and re-open your current shell. <==

If you'd prefer that conda's base environment not be activated on startup,
set the auto_activate_base parameter to false:

conda config --set auto_activate_base false

Thank you for installing Miniconda3!
```

미니콘다 설치 후

source ~/.bashrc 로 conda 환경으로 진입한다. or wsl 을 재시작 한다.

wsl -t 배포판 이름 -> wsl -d 배포판 이름 or 검색 -> Ubuntu -> 실행

conda -V

```
(base) [redacted]~$ conda -V  
conda 23.5.2
```

정상설치 되었다면 버전이 나타난다.

conda 환경을 생성한다.

공식 매뉴얼에서는 3.9 버전을 소개하고 있으나 3.10 을 추천한다. (Colab 호환)

conda create --name tf python=3.9

conda create --name tf python=3.10 (취사선택)

```
Proceed ([y]/n)? y
```

```
Downloading and Extracting Packages
```

```
Preparing transaction: done
```

```
Verifying transaction: done
```

```
Executing transaction: done
```

```
#
```

```
# To activate this environment, use
```

```
#
```

```
#     $ conda activate tf
```

```
#
```

```
# To deactivate an active environment, use
```

```
#
```

```
#     $ conda deactivate
```

conda deactivate 로 base 에서 나가고

conda activate tf 로 tf 로 들어간다.

```
(base) [redacted]~$ conda deactivate  
[redacted]~$ conda activate tf  
(tf) [redacted]~$
```

nvidia-smi 로 gpu 정보를 확인한다.

```
(tf) [redacted]~$ nvidia-smi
Wed Sep  6 01:12:26 2023
```

NVIDIA-SMI 535.103				Driver Version: 537.13			CUDA Version: 12.2		
GPU	Name	Perf	Persistence-M	Bus-Id	Disp.A	Memory-Usage	Volatile GPU-Util	Uncorr. Compute M.	ECC MIG M.
Fan	Temp		Pwr:Usage/Cap						
0	NVIDIA GeForce RTX 3090	P5	On	00000000:0B:00.0	On	2098MiB / 24576MiB	29%	Default	N/A
50%	49C		49W / 420W	2098MiB / 24576MiB					N/A

Processes:							GPU Memory Usage	
GPU	GI ID	CI ID	PID	Type	Process name			
0	N/A	N/A	23	G	/Xwayland			N/A

conda install -c conda-forge cudatoolkit=11.8.0 로 cudatoolkit 을 설치한다.

(11.8 을 설치하는 이유는 conda 에선 현재 11.8 이 최신이기 때문.)

```
Proceed ([y]/n)? y

Downloading and Extracting Packages

Preparing transaction: done
Verifying transaction: done
Executing transaction: | By downloading and using the CUDA Toolkit conda packages, you accept the terms and conditions of the CUDA End User License Agreement (EULA): https://docs.nvidia.com/cuda/eula/index.html
done
```

pip install nvidia-cudnn-cu11==8.6.0.163 로 cuDNN 을 설치한다. (버전 사유는 상동)

```
(tf) [redacted]~$ pip install nvidia-cudnn-cu11==8.6.0.163
Collecting nvidia-cudnn-cu11==8.6.0.163
  Downloading nvidia_cudnn_cu11-8.6.0.163-py3-none-manylinux1_x86_64.whl (715.7 MB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 715.7/715.7 MB 8.6 MB/s eta 0:00:00
Collecting nvidia-cublas-cu11 (from nvidia-cudnn-cu11==8.6.0.163)
  Downloading nvidia_cublas_cu11-11.11.3.6-py3-none-manylinux1_x86_64.whl (417.9 MB)
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 417.9/417.9 MB 12.6 MB/s eta 0:00:00
Installing collected packages: nvidia-cublas-cu11, nvidia-cudnn-cu11
Successfully installed nvidia-cublas-cu11-11.11.3.6 nvidia-cudnn-cu11-8.6.0.163
```

```
CUDNN_PATH=$(dirname $(python -c "import nvidia.cudnn;print(nvidia.cudnn.__file__)"))
```

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CUDNN_PATH/lib:$CONDA_PREFIX/lib/
```

위 커맨드를 한줄씩 실행한다.

```
(tf) [redacted]~$ CUDNN_PATH=$(dirname $(python -c "import nvidia.cudnn;print(nvidia.cudnn.__file__)"))
(tf) [redacted]~$ export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CUDNN_PATH/lib:$CONDA_PREFIX/lib/
```

```
mkdir -p $CONDA_PREFIX/etc/conda/activate.d
```

```
echo 'CUDNN_PATH=$(dirname $(python -c "import
nvidia.cudnn;print(nvidia.cudnn.__file__)"))' >>
$CONDA_PREFIX/etc/conda/activate.d/env_vars.sh
```

```
echo 'export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CUDNN_PATH/lib:$CONDA_PREFIX/lib/'
>> $CONDA_PREFIX/etc/conda/activate.d/env_vars.sh
```

위 커맨드를 하나씩 실행한다.

```
(tf) [redacted] $ mkdir -p $CONDA_PREFIX/etc/conda/activate.d
(tf) [redacted] $ echo 'CUDNN_PATH=$(dirname $(python -c "import nvidia.cudnn;print(nvidia.cudnn.__file__)"))' >
> $CONDA_PREFIX/etc/conda/activate.d/env_vars.sh
(tf) [redacted] $ echo 'export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CUDNN_PATH/lib:$CONDA_PREFIX/lib/' >> $CONDA_PR
EFIX/etc/conda/activate.d/env_vars.sh
```

pip install --upgrade pip 로 pip 업데이트 한다.

pip install tensorflow==2.13.0 로 tensorflow 2.13.0 버전을 설치한다.

환경이나 상황에 따라 버전은 취사선택 가능하며, 본 예시에서는 최신버전으로 설치.

테스트 코드를 실행한다.

```
CPU Ver TEST -> python3 -c "import tensorflow as tf;
print(tf.reduce_sum(tf.random.normal([1000, 1000])))"
```

```
2023-09-06 01:23:26.046750: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1639] Created device /job:localhost/repli
ca:0/task:0/device:GPU:0 with 21598 MB memory: -> device: 0, name: NVIDIA GeForce RTX 3090, pci bus id: 0000:0b:00.0, c
ompute capability: 8.6
tf.Tensor(-1044.2317, shape=(), dtype=float32)
```

```
GPU Ver TEST -> python3 -c "import tensorflow as tf;
print(tf.config.list_physical_devices('GPU'))"
```

```
Your kernel may have been built without NUMA support.
[PhysicalDevice(name='/physical_device:GPU:0', device_type='GPU')]
```

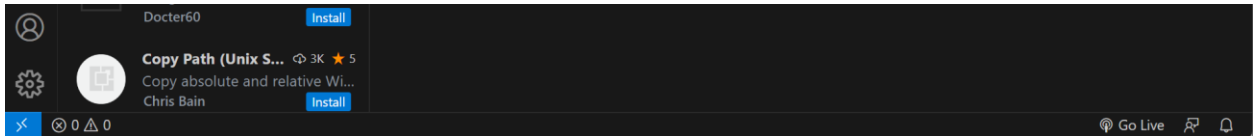
CPU 에서 Tensor 가 Return 되고

GPU 에서 GPU 목록이 Return 되면 정상 설치된 것이다.

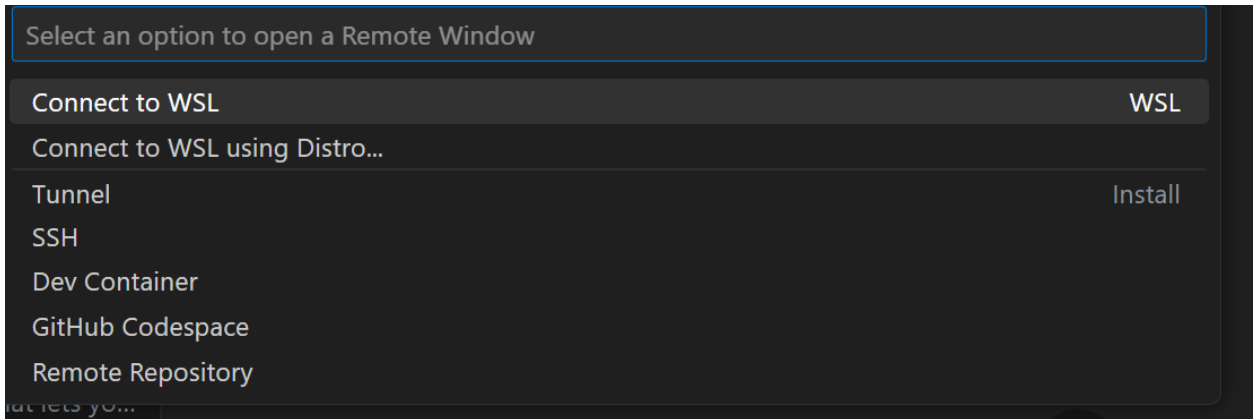
5. VSCode 연동

VSCode 실행 -> Extensions -> Remote-WSL 또는 WSL 검색 -> 설치

설치 후 VSCode 종료 후 재실행 해 보면

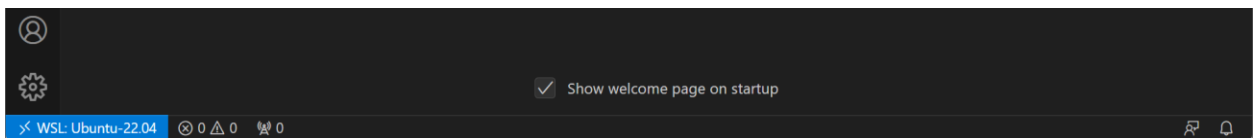


좌측하단부에 >< 아이콘을 확인할 수 있다.



해당 버튼을 눌러 WSL 실행 시키거나

Linux 터미널에서 `code .` 를 실행시키면 자동으로 연결된다.



연결되면 하단 Remote Tab 에 배포판 이름이 나타난다.

GPU Windows Native

Tensorflow 2.11 부터는 Windows Native 지원이 끊기고 WSL 로 이전되어

Native 설치가 필요한 경우 2.10 버전으로 내려 설치해야 한다.

Driver 설치는 GPU 설치를 참조한다.

(선택) TensorRT https://docs.nvidia.com/deeplearning/tensorrt/archives/index.html#trt_7

TensorRT 설치 시 성능개선을 기대할 수 있으나 필수사항은 아니다.

1. Driver 설치

<https://developer.nvidia.com/cuda-toolkit>

위 페이지에서 Cuda Toolkit 을 설치합니다.

Download Now -> Select your environment(Windows in this case) -> Choose the Installer Type

Operating System	Linux	Windows		
Architecture	x86_64			
Version	10	11	Server 2019	Server 2022
Installer Type	exe (local)	exe (network)		

Local 은 package 전체, network 는 다운로더만 받아옵니다.

특정 버전으로 맞추고 싶은 경우 Download Page 하단 Resources -> Archive of Pre...로 이동하여 버전을 선택 후 받는다.

Resources

- [CUDA Documentation/Release Notes](#)
- [MacOS Tools](#)
- [Training](#)
- [Sample Code](#)
- [Forums](#)
- [Archive of Previous CUDA Releases](#)
- [FAQ](#)
- [Open Source Packages](#)
- [Submit a Bug](#)
- [Tarball and Zip Archive Deliverables](#)

CUDA Toolkit Archive

Home

Previous releases of the CUDA Toolkit, GPU Computing SDK, documentation and developer drivers can be found using the links below. Please select the release you want from the list below, and be sure to check www.nvidia.com/drivers for more recent production drivers appropriate for your hardware configuration.

[Download Latest CUDA Toolkit](#)

[Learn More about CUDA Toolkit](#)

Latest Release


[CUDA Toolkit 12.2.2 \(August 2023\)](#), [Versioned Online Documentation](#)

Archived Releases

[CUDA Toolkit 12.2.1 \(July 2023\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 12.2.0 \(June 2023\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 12.1.1 \(April 2023\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 12.1.0 \(February 2023\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 12.0.1 \(January 2023\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 12.0.0 \(December 2022\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 11.8.0 \(October 2022\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 11.7.1 \(August 2022\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 11.7.0 \(May 2022\)](#), [Versioned Online Documentation](#)
[CUDA Toolkit 11.6.2 \(March 2022\)](#), [Versioned Online Documentation](#)

Download Installer for Windows 11 x86_64

The base installer is available for download below.

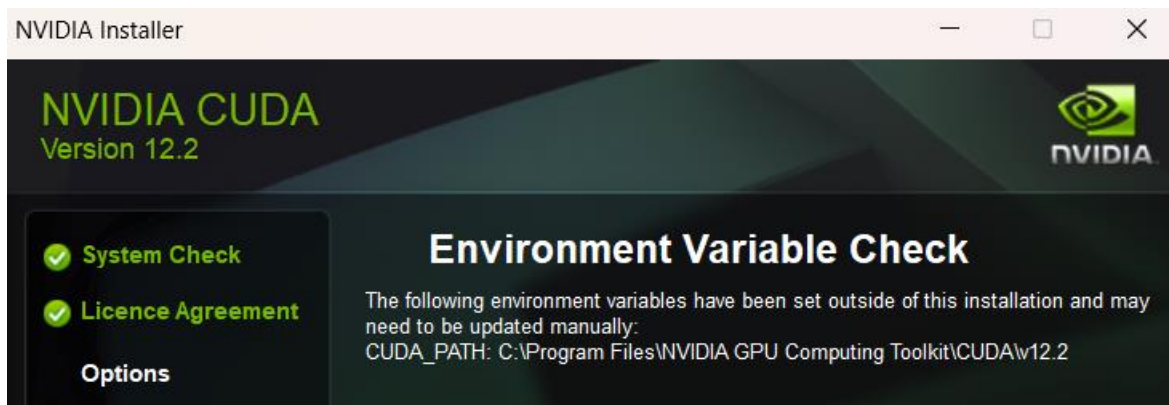
Base Installer	Download (3.0 GB) 
----------------	---

Installation Instructions:

1. Double click cuda_12.2.2_537.13_windows.exe
2. Follow on-screen prompts

The checksums for the installer and patches can be found in [Installer Checksums](#).
For further information, see the [Installation Guide for Microsoft Windows](#) and the [CUDA Quick Start Guide](#).

Download 를 눌러 Installer 를 받고 설치한다, 통상 Driver 처럼 진행하면 된다.



환경변수 설정을 별도로 요구하는 경우 설치 완료 후 해당 환경변수를 추가/수정 한다.

2. cuDNN 설치

<https://developer.nvidia.com/cudnn>

위 링크에서 Download cuDNN 으로 이동한다.

NVIDIA 계정이 필수이므로 가입/로그인 후 진행한다.

cuDNN Download

NVIDIA cuDNN is a GPU-accelerated library of primitives for deep neural networks.

☒ **I Agree To the Terms of the cuDNN Software License Agreement**

Note: Please refer to the Installation Guide for release prerequisites, including supp

For more information, refer to the cuDNN Developer Guide, Installation Guide and R

Download cuDNN v8.9.4 (August 8th, 2023), for CUDA 12.x

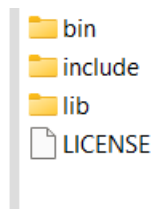
Download cuDNN v8.9.4 (August 8th, 2023), for CUDA 11.x

Archived cuDNN Releases

약관 동의 후 설치된 CUDA 버전과 환경에 맞게 받는다. (구버전 필요 시 Archived 로 이동)

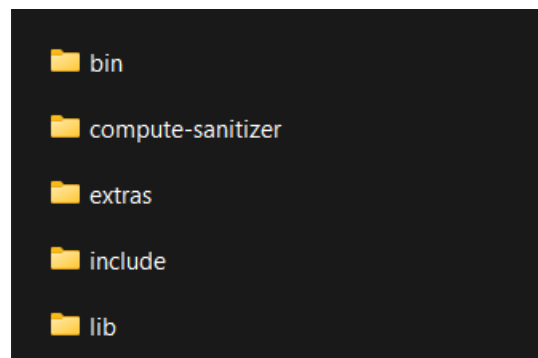
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA 로 이동하여(기타 경로 미 지정 시)

자신이 설치한 CUDA 버전으로 이동한 다음



cuDNN 압축폴더 내 폴더 3 개를

CUDA 설치경로에 붙여넣는다(덮어쓰기)



```
PS C:\Users\ [redacted] > nvidia-smi
Wed Sep  6 00:16:03 2023

+-----+
| NVIDIA-SMI 537.13              | Driver Version: 537.13          | CUDA Version: 12.2             |
+-----+-----+-----+-----+-----+-----+
| GPU  Name      TCC/WDDM    | Bus-Id      Disp.A    | Volatile Uncorr. ECC |
| Fan  Temp      Perf      Pwr:Usage/Cap |      Memory-Usage | GPU-Util  Compute M. |
|=====+=====+=====+=====+=====+=====+
|   0  NVIDIA GeForce RTX 3090  WDDM    | 00000000:0B:00.0 On |         N/A         |
| 49%   49C     P5             50W / 420W | 1779MiB / 24576MiB |      31%    Default |
|                                     |                     |         N/A         |
+-----+-----+-----+-----+-----+-----+

Processes:
+-----+-----+-----+-----+-----+-----+
| GPU  GI   CI          PID   Type   Process name                      | GPU Memory |
|   ID  ID   ID                                     |      Usage |
+-----+-----+-----+-----+-----+-----+

```

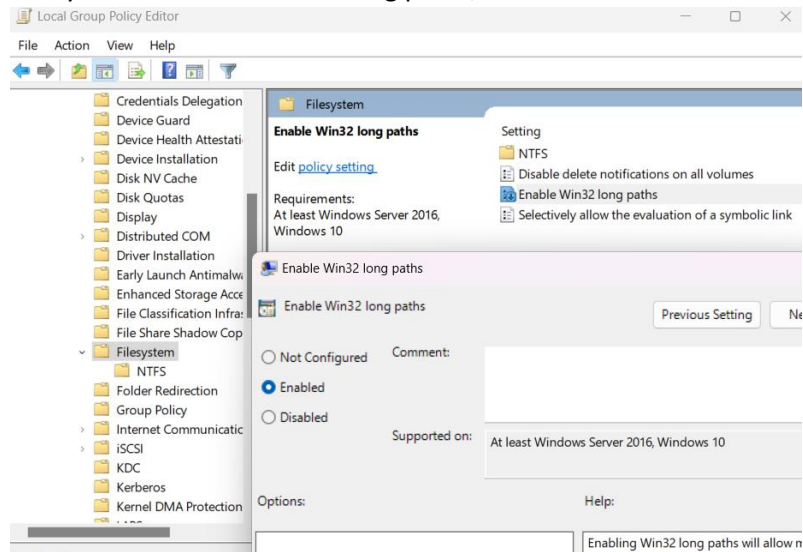
재부팅후 nvidia-smi 로 gpu 드라이버 상태 확인 후

```
PS C:\Users\ [redacted] > nvcc --version
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2023 NVIDIA Corporation
Built on Tue_Aug_15_22:09:35_Pacific_Daylight_Time_2023
Cuda compilation tools, release 12.2, V12.2.140
Build cuda_12.2.r12.2/compiler.33191640_0
```

Nvcc --version 으로 cuda 버전을 확인한다.

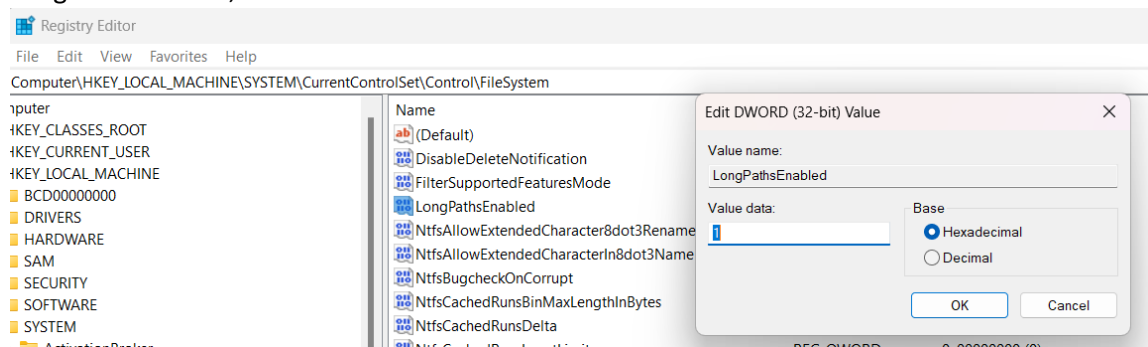
3. long path Activate

gpedit.msc -> Local Computer Policy -> Computer Config -> Admin Templates -> System -> FileSystem -> Enable Win32 long paths, set Enable



Local Group Policy Editor 가 없는 경우

regedit -> HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\FileSystem -> LongPathsEnabled, set 1



4. Miniconda or Anaconda Prompt Open
Powershell Prompt Recommended

```
conda create --name tf python=3.9 (or 3.10)
```

```
conda deactivate  
conda activate tf
```

```
pip install "tensorflow<2.11" or tensorflow==2.10.0 (or 2.10.1)
```

5. 확인

```
CPU Test -> python -c "import tensorflow as tf; print(tf.reduce_sum(tf.random.normal([1000, 1000])))"
```

```
GPU Test -> python -c "import tensorflow as tf; print(tf.config.list_physical_devices('GPU'))"
```

CPU Test 에서 Tensor Return and GPU Test 에서 GPU list Return 되면 완료.

CPU 설치

1. 설치

conda 환경 생성

conda create --name tf python=3.9 (or 3.10)

conda 환경 전환

conda deactivate

conda activate tf

python or conda 에서

python3 -m pip install tensorflow 또는

python -m pip install tensorflow

(pip install tensorflow 도 가능)

(만약 실행이 안되는 경우라면 Python 설치여부 또는 환경변수를 확인하여

자신의 환경변수에 설정된 이름대로 수정해서 실행하시오.)

2. 확인

python3 -c "import tensorflow as tf; print(tf.reduce_sum(tf.random.normal([1000, 1000])))"

또는

python -c "import tensorflow as tf; print(tf.reduce_sum(tf.random.normal([1000, 1000])))"

Tensor 가 Return 시 완료.

VSCode 에서 실행 확인하기

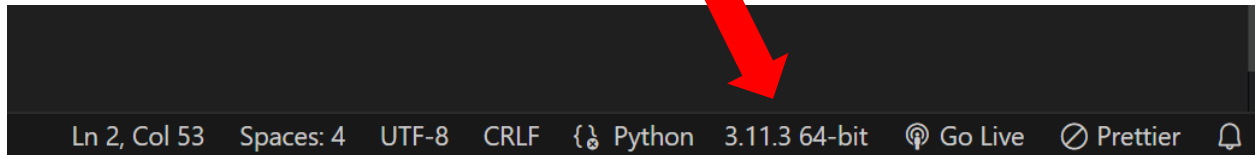
1. CPU

VSCode 를 실행하여

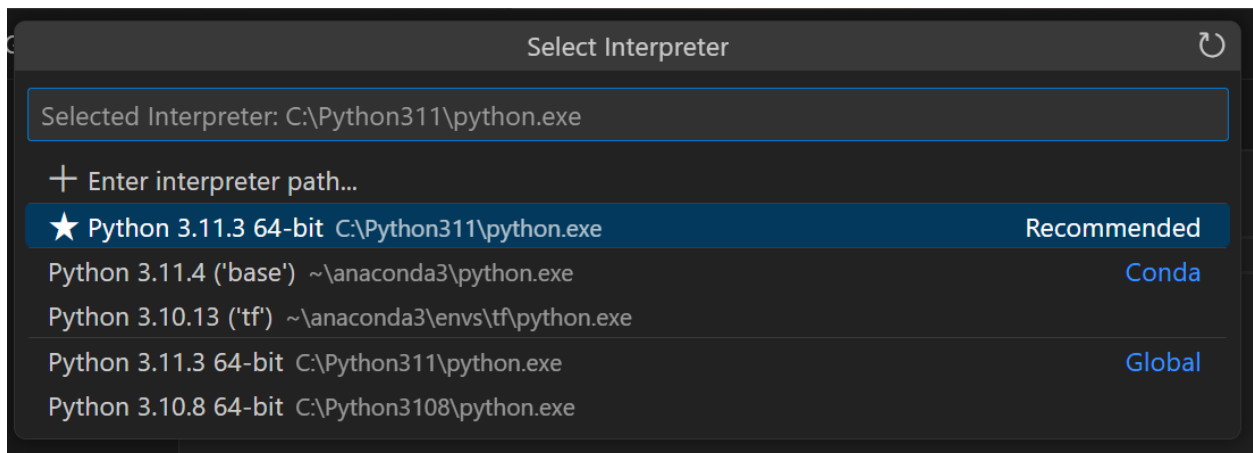
작업폴더 열기 -> 임의 py script 생성 후

아래 코드를 입력한다.

```
import tensorflow as tf  
print(tf.reduce_sum(tf.random.normal([1000, 1000])))
```



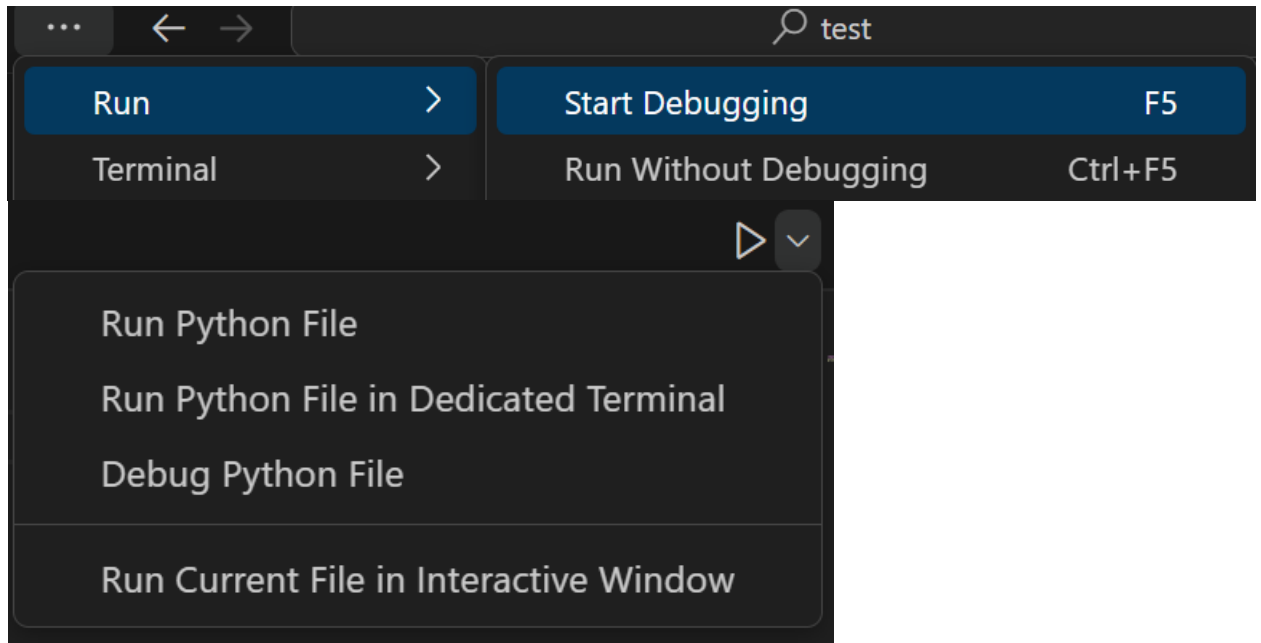
우측하단 Python 실행환경을 누른다



자신이 실행할 환경을 선택한다. (tensorflow 가 설치되어 있음 and 실행할 환경)

여기서는 tf 환경을 선택하여 진행한다.

(표시되는 버전 관련해서는 본인의 환경구축에 따라서 판이할 수 있음.)



외부 Terminal, VSCode 실행 등.. 어떤 방법으로든 Code 를 실행하여 결과를 확인한다.

```
tf.Tensor(-1055.2622, shape=(), dtype=float32)
```

CPU 의 경우 Tensor 가 반환되면 성공

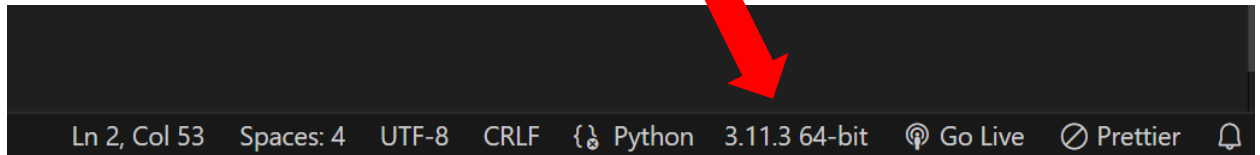
2. GPU-Native

VSCode 를 실행하여

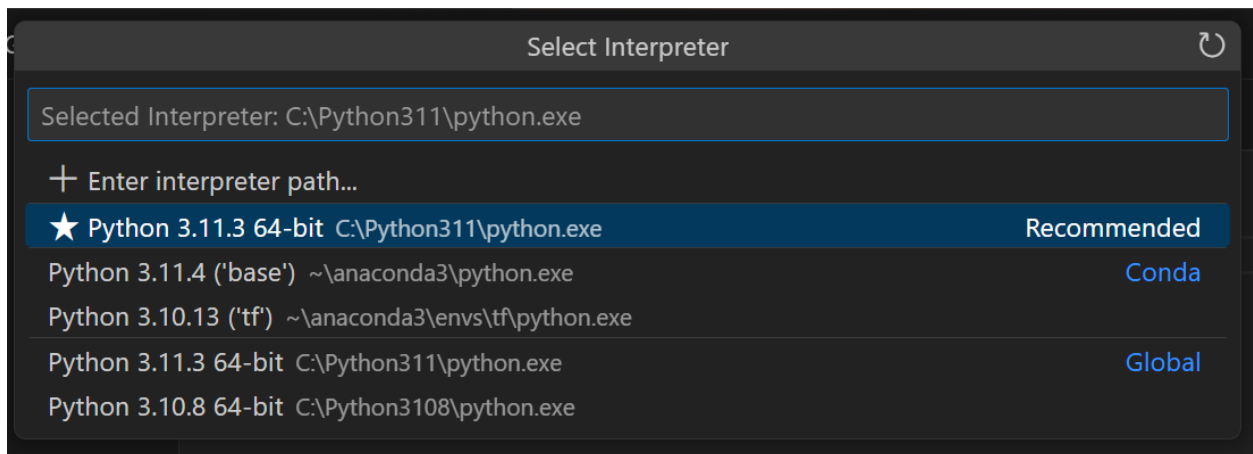
작업폴더 열기 -> 임의 py script 생성 후

아래 코드를 입력한다.

```
import tensorflow as tf  
print(tf.config.list_physical_devices('GPU'))
```



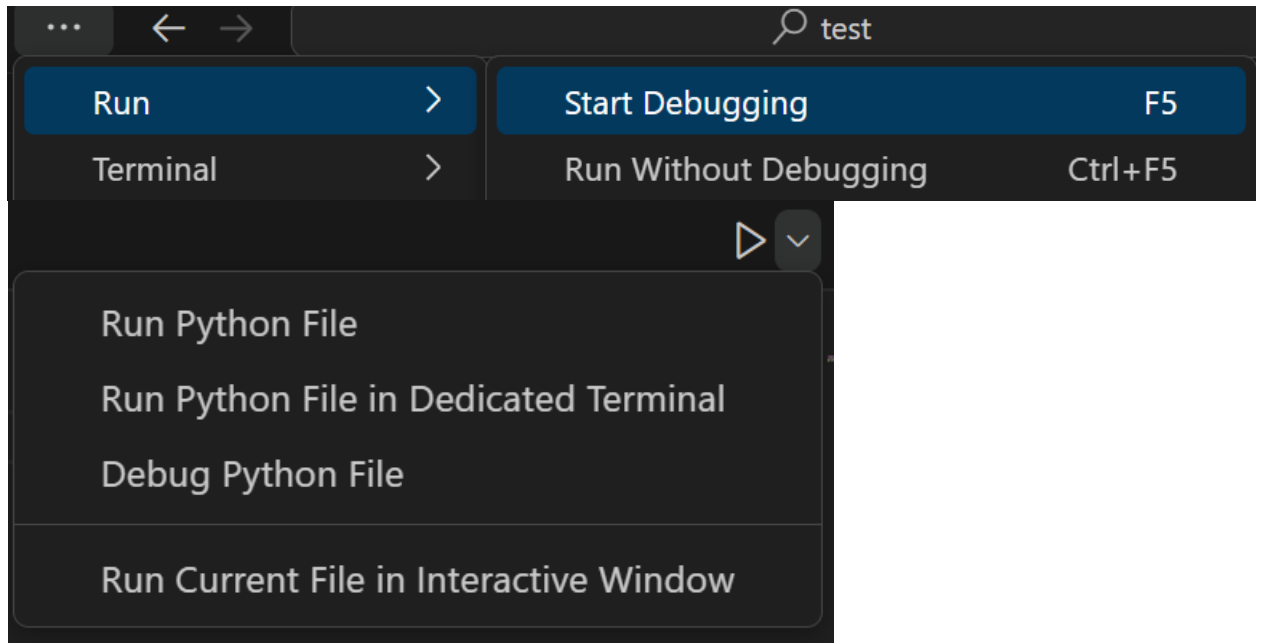
우측하단 Python 실행환경을 누른다



자신이 실행할 환경을 선택한다. (tensorflow 가 설치되어 있음 and 실행할 환경)

여기서는 tf 환경을 선택하여 진행한다.

(표시되는 버전 관련해서는 본인의 환경구축에 따라서 판이할 수 있음.)



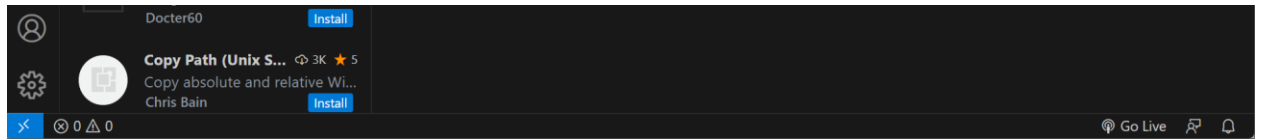
외부 Terminal, VSCode 실행 등.. 어떤 방법으로든 Code 를 실행하여 결과를 확인한다.

```
[PhysicalDevice(name='/physical_device:GPU:0', device_type='GPU')]
```

GPU 의 경우 Device List 가 반환되면 성공

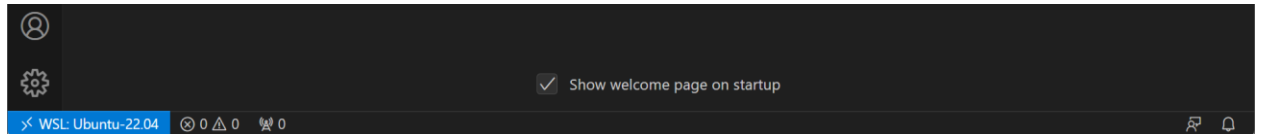
3. GPU-WSL

VSCode 를 실행하여



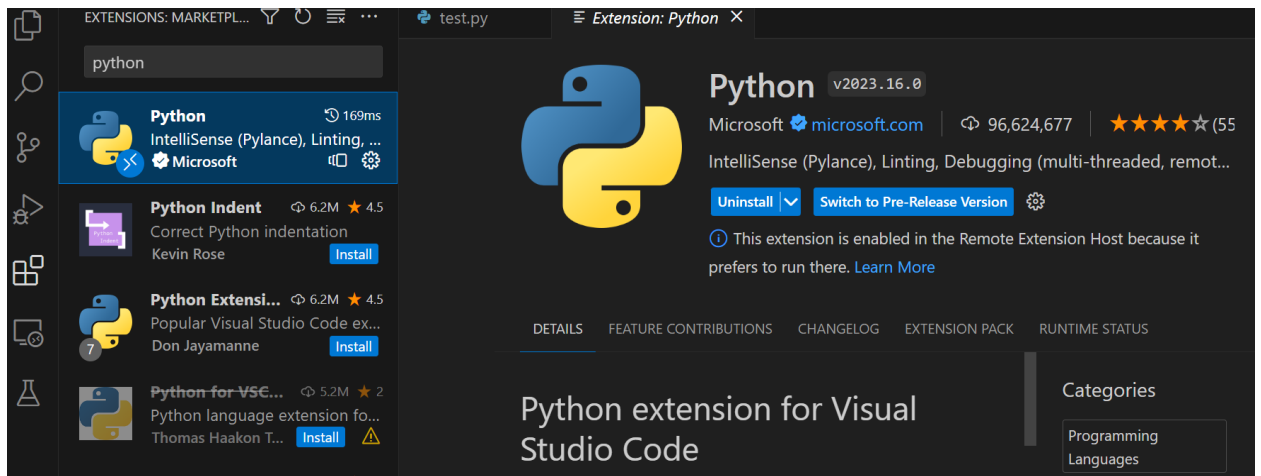
좌측하단부에 >< 버튼을 누르거나

Linux 터미널에서 `code .` 를 입력해 WSL 환경에 연결한다.



연결되면 하단 Remote Tab 에 배포판 이름이 나타난다.

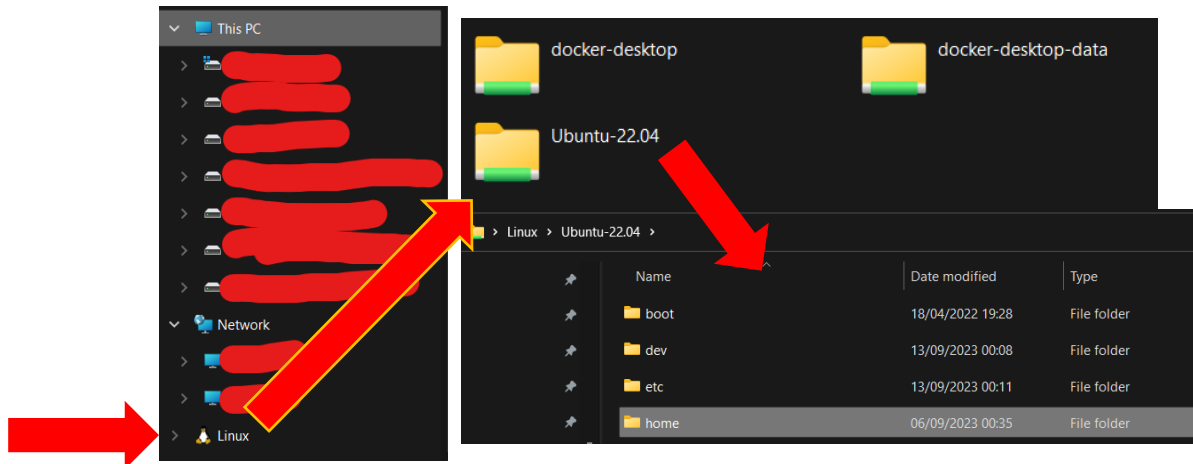
(주의)



WSL Remote 환경에서는 Host 와는 별개의 공간이므로 일부 Extension 은 사용할 수 없을 수 있으며, Host 의 Extension 이 설치되어 있지 않으므로 필요한 것을 설치해 준다.

Python 이 설치되어 있지 않은 경우 선택화면이나 관련기능을 찾을 수 없다.

WSL 에서 파일 접근하기 (2 기준 설명이므로 기타버전은 검색하시기 바랍니다.)



(자신이 설치한 환경의 이름만 나타납니다, 각자 환경에 따라 상이할 수 있음)

Explorer -> Linux -> [Distribution you installed] -> Linux Storage

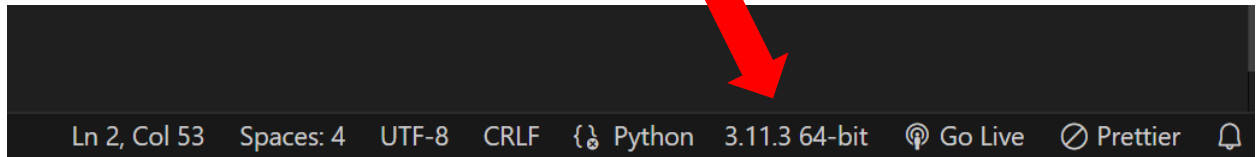
User (Home)Directory -> /home/[YOUR USER NAME]/

or 터미널에서 조작.

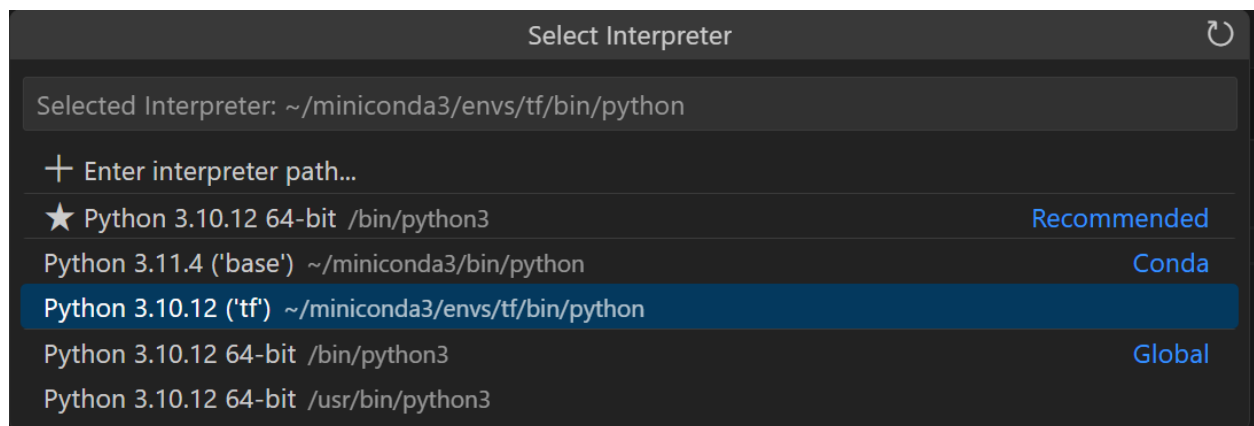
작업폴더 열기 -> 임의 py script 생성 후

아래 코드를 입력한다.

```
import tensorflow as tf  
print(tf.config.list_physical_devices('GPU'))
```



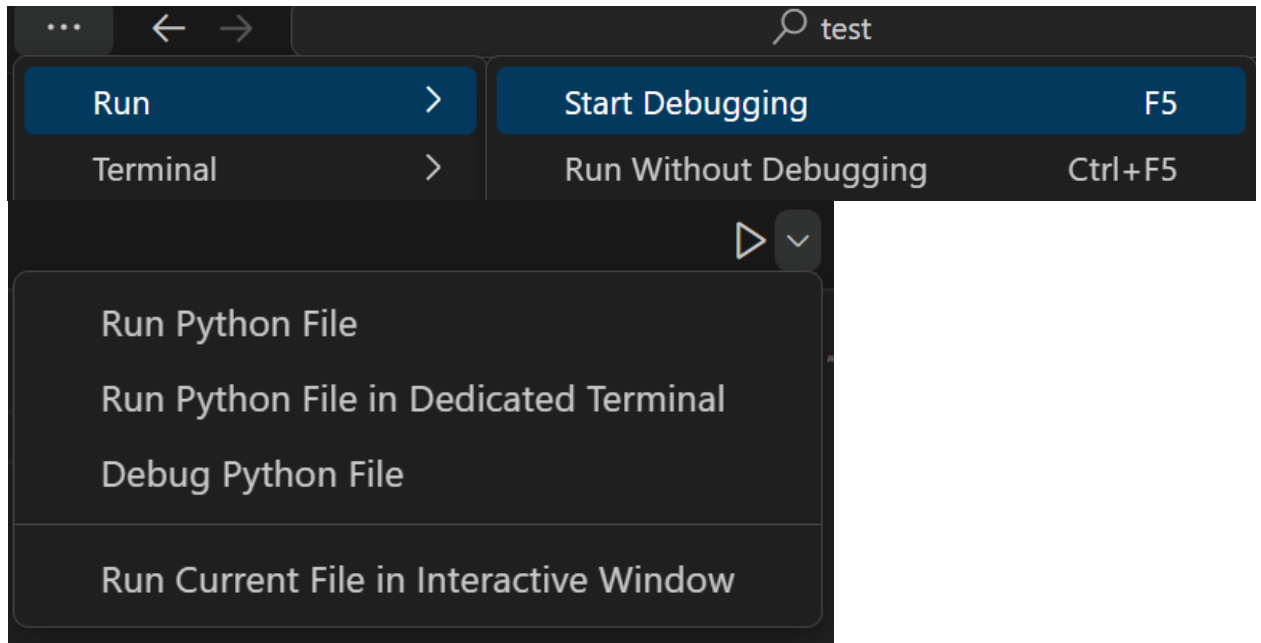
우측하단 Python 실행환경을 누른다



자신이 실행할 환경을 선택한다. (tensorflow 가 설치되어 있음 and 실행할 환경)

여기서는 tf 환경을 선택하여 진행한다.

(표시되는 버전 관련해서는 본인의 환경구축에 따라서 판이할 수 있음.)



외부 Terminal, VSCode 실행 등.. 어떤 방법으로든 Code 를 실행하여 결과를 확인한다.

```
[PhysicalDevice(name='/physical_device:GPU:0', device_type='GPU')]
```

GPU 의 경우 Device List 가 반환되면 성공

P.S.

tensorflow 설치 시 일부 패키지가 Dependency 로 인하여 설치도중 오류가 발생할 수 있다.

pip uninstall 로 삭제 후

설치 진행중 오류 발생한 패키지를 별도로 pip install [Package Name](==[Version])

개별설치 하고 다시 설치를 진행한다.

wsl 이 실행되고 있는 중에는 VM 이 실행되고 있으므로 메모리를 많이 점유하고 있으며

HyperVisor 버그로 한번 메모리가 할당되면 VM 종료 전까지 반환하지 않을 수 있다.

메모리가 충분한 PC 에서 진행하는 것이 좋으며 (최소 16G, 권장 32G, 추천 64~128G)

작업 종료 후 VM 을 종료하는 것이 좋다. (wsl -t [Distro Name])