

# pipenv system GPU利用版

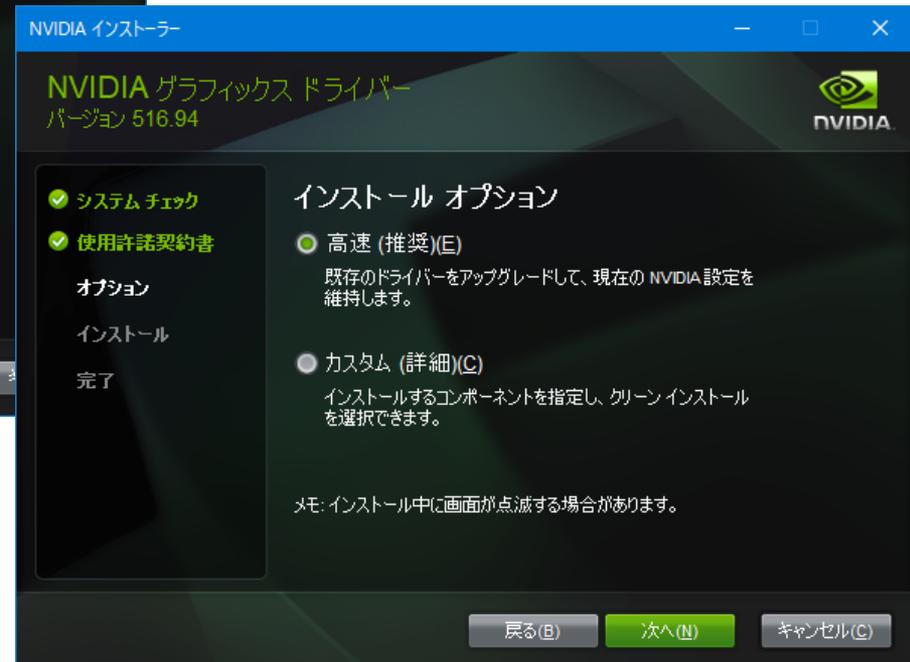
First ed. October 07, 2021

Updated Jan. 29, 2025

By H.Nishiyama, Niigata-Univ-Dent.

# GPUのドライバを最新版にしておく

- <https://www.nvidia.com/ja-jp/drivers/>



# 搭載されているGPUに応じて 制約を確認する

- 搭載されているGPUが利用可能か確認する  
(NVIDIAの場合)
  - <https://developer.nvidia.com/cuda-gpus>



- ドライバのバージョンにて利用可能なCUDAのバージョンをチェックする。
  - <https://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html>

Table 2: CUDA Toolkit and Minimum Required Driver Version for CUDA Minor Version Compatibility

CUDA Toolkit	Minimum Required Driver Version for CUDA Minor Version Compatibility*	
	Linux x86_64 Driver Version	Windows x86_64 Driver Version
CUDA 12.x	>=525.60.13	>=528.33
CUDA 11.8.x CUDA 11.7.x CUDA 11.6.x CUDA 11.5.x CUDA 11.4.x CUDA 11.3.x CUDA 11.2.x CUDA 11.1.x	>=450.80.02	>=452.39
CUDA 11.0 (11.0.3)	>=450.36.06**	>=451.22**

\* Using a Minimum Required Version that is **different** from Toolkit Driver Version could be allowed in compatibility mode – please read the CUDA Compatibility Guide for details.

\*\* CUDA 11.0 was released with an earlier driver version, but by upgrading to Tesla Recommended Drivers 450.80.02 (Linux) / 452.39 (Windows), minor version compatibility is possible across the CUDA 11.x family of toolkits.

NVIDIAのドライババージョンから、  
利用可能なCUDAを確認する。  
※予めNVIDIAのドライバは最新の  
ものに更新しておくこと。

Table 3: CUDA Toolkit and Corresponding Driver Versions

CUDA Toolkit	Toolkit Driver Version	
	Linux x86_64 Driver Version	Windows x86_64 Driver Version
CUDA 12.8 GA	>=570.26	>=570.65
CUDA 12.6 Update 3	>=560.35.05	>=561.17
CUDA 12.6 Update 2	>=560.35.03	>=560.94
CUDA 12.6 Update 1	>=560.35.03	>=560.94
CUDA 12.6 GA	>=560.28.03	>=560.76
CUDA 12.5 Update 1	>=555.42.06	>=555.85
CUDA 12.5 GA	>=555.42.02	>=555.85
CUDA 12.4 Update 1	>=550.54.15	>=551.78

# TensorFlowかPyTorchか？

必要なPythonやCUDAのバージョンが異なるので注意

- PyTorch
  - 数年前から利用率がTensorFlowを上回り、テキストおよびネット上のサンプルも増えてきている。
  - APIが低レベルのため、プログラミングの難易度が高くなる傾向あり、pipenv環境へのインストールも難しくなっていたが、ここ1, 2年で劇的に使い勝手が良くなった。
  - CUDAおよびcuDNNが一体化しており、OSレベルでインストールされたバージョンとは無関係なので、pipenv環境などでも使いやすい。
- TensorFlow
  - 現行利用している教科書と演習課題の一部にて使用している。
  - TensorFlow1の時代と比較し2の時代のものは内部構造が理解しにくくなっている。
  - CUDAおよびcuDNNがシステムにインストールされていないと使えないため、Docker使うか、システム(OS)レベルでの切り換えが必要
  - Tensorflowを含めライブラリのバージョンが上がっていくと、古いソフトが動かないことが多いので、Docker等の利用を考えた方が良いと思われる。

# NVIDIA-GPUをpipenv systemで使う

- 適するcuDNN, Python および ソフトの組み合わせバージョンを確認(重要)
  - PyTorchの場合は  
<https://pytorch.org/get-started/locally/>
  - TensorFlowは  
[https://www.tensorflow.org/install/source\\_windows?hl=ja#gpu](https://www.tensorflow.org/install/source_windows?hl=ja#gpu)
- TensorFlowでは適切なCUDA toolkit と cuDNN を別にダウンロードして、インストールする

# PyTorchの場合

Pipenv利用時、まずはPipとして適合システム確認※

- PyTorchのサイト: <https://pytorch.org/>
  - 上記内から下記のインストールに入り、自身のシステムを確認
  - <https://pytorch.org/get-started/locally/>
  - CUDAを含んだライブラリとしてインストールされる

**NOTE:** Latest PyTorch requires Python 3.9 or later.

PyTorch Build	Stable (2.5.1)		Preview (Nightly)	
Your OS	Linux	Mac	Windows	
Package	Conda	Pip	LibTorch	Source
Language	Python		C++ / Java	
Compute Platform	CUDA 11.8	CUDA 12.1	CUDA 12.4	ROCm 6.2
Run this Command:	<pre>pip3 install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu121</pre>			

例えばpython3.9以上でpytorch2.5.1でwindowsでCUDA 12.1

※Pipenvの場合、Pipでのインストールとは異なるので注意。下記参照  
<https://qiita.com/aujinen/items/db919ceee6da8db1155c>  
<https://www5.dent.niigata-u.ac.jp/~nisiyama/grad/Pipenv-PyTorch.pdf>

# TensorFlowの場合(ここから以降)

[https://www.tensorflow.org/install/source\\_windows?hl=ja#gpu](https://www.tensorflow.org/install/source_windows?hl=ja#gpu)

TensorFlow

インストール 学ぶ API リソース もっと見る

検索 / 日本語

GitHub ログイン

GPU

★ 注: ネイティブ Windows での GPU サポートは、TF 2.11 以降、2.10 以前のバージョンでのみ利用可能です。Windows では CUDA ビルドはサポートされていません。Windows で TensorFlow GPU を使用するには、WSL2 で TensorFlow をビルド/インストールするか、TensorFlow-DirectML-Plugin で tensorflow-cpu を使用する必要があります。

バージョン	Pythonのバージョン	コンパイラ	ビルドツール	クドン	CUDA
tensorflow_gpu-2.10.0	3.7~3.10	MSVC 2019	バazel5.1.1	8.1	11.2
tensorflow_gpu-2.9.0	3.7~3.10	MSVC 2019	バazel5.0.0	8.1	11.2
tensorflow_gpu-2.8.0	3.7~3.10	MSVC 2019	バazel5.1	8.1	11.2
tensorflow_gpu-2.7.0	3.7-3.9	MSVC 2019		8.1	11.2
tensorflow_gpu-2.6.0	3.6-3.9	MSVC 2019		8.1	11.2
tensorflow_gpu-2.5.0	3.6-3.9	MSVC 2019		8.1	11.2
tensorflow_gpu-2.4.0	3.6-3.8	MSVC 2019		8.0	11.0

ソースからビルドする

Linux / macOS

Windows

SIG Build

言語バインディング

Java

Java (旧)

C

Go

TensorFlow ソースコードをダウンロードする

オプション: 環境変数の設定

オプション: ビルドを構成する

pip パッケージをビルドしてインストールする

パッケージビルダーをビルドする

パッケージをビルドする

パッケージをインストールする

MSYS シェルを使用してビルドする

テストされたビルド構成

GPU

GPU

Windows 用のセットアップ

Python と TensorFlow パッケージの依存関係をインストールする

例えば、この設定にする場合

Pythonは3.7~3.10

Microsoft C++ライブラリには2019

cuDNN 8.1

CUDA 11.2

以上の環境にてTensorflow-gpuの2.10.0が使えるということ

# TensorFlowの場合

## Python, CUDA, cuDNNのバージョン

2024/03/02現在・稼働確認済みのもの

Python	CUDA toolkit	cuDNN	TensorFlow-GPU
3.5-3.7	10.1	7.65	1.14.0
3.6-3.9	11.2以上	8.1以上	2.6.0
3.7-3.10	11.2以上	8.1以上	2.8.0以上(※)

※Python3.10.7, CUDA11.7.1, cuDNN8.6.0, TensorFlow-GPU2.10にて確認済み

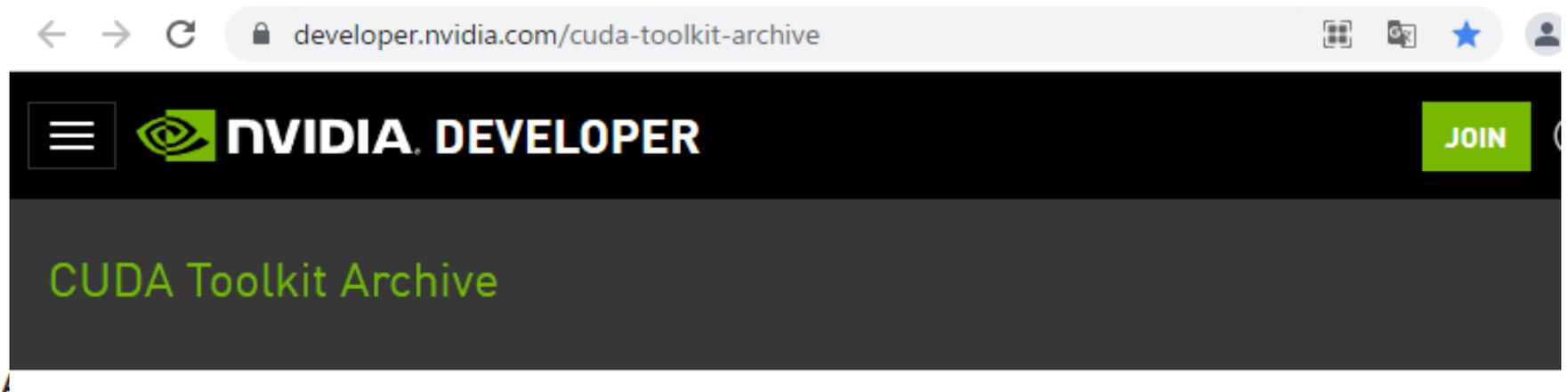
★ **Note:** GPU support on native-Windows is only available for 2.10 or earlier versions, starting in TF 2.11, CUDA build is not supported for Windows. For using TensorFlow GPU on Windows, you will need to build/install TensorFlow in WSL2 or use tensorflow-cpu with TensorFlow-DirectML-Plugin

注:ネイティブ Windows での GPU サポートは、2.10 以前のバージョンでのみ利用可能です。TF 2.11 以降、Windows では CUDA ビルドはサポートされていません。Windows で TensorFlow GPU を使用するには、WSL2 で TensorFlow をビルド/インストールするか、TensorFlow-DirectML-Plugin で tensorflow-cpu を使用する必要があります。

# TensorFlowの場合

CUDAをcuda-toolkit-archiveからインストールする

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



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](#)

CUDA Toolkit 11.6.1 (February 2022), [Versioned Online Documentation](#)

CUDA Toolkit 11.6.0 (January 2022), [Versioned Online Documentation](#)

CUDA Toolkit 11.5.2 (February 2022), [Versioned Online Documentation](#)

CUDA 11.7.1をインストールする場合  
※通常、2世代前位が安定している。

## 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

Architecture

Version

Installer Type

Windows10の場合

Linux	Windows			
x86_64				
10	11	Server 2016	Server 2019	Server 2022
exe (local)	exe (network)			

## Download Installer for Windows 10 x86\_64

The base installer is available for download below.

## &gt; Base Installer

Installation Instructions:

1. Double click cuda\_11.7.1\_windows\_network.exe
2. Follow on-screen prompts

Download (34.5 MB)



NVIDIA インストーラー

## NVIDIA CUDA

バージョン 11.7

- ✓ システムチェック
- ✓ 使用許諾契約書
- オプション
- インストール
- 完了

### インストール オプション

高速 (推奨)(E)  
Installs all CUDA components and overwrites current Display Driver.

カスタム (詳細)(C)  
Allows you to select the components

メモ: インストール中に画面が点滅する場合があります。

戻る(B) 次へ(N) キャンセル(C)

最初にインストールする場合や、複数インストールしていても、最新版をインストールする場合「高速(推奨)」を選択。

ダウングレード版(バージョンの小さなもの)を追加でインストールする場合、「カスタムインストール」

# カスタムインストールの場合

NVIDIA インストーラー

NVIDIA CUDA  
バージョン 11.2

システムチェック  
使用許諾契約書

オプション  
インストール  
完了

**カスタム インストール オプション**

ドライバー コンポーネントの選択:

コンポーネント	新しいバー...	現在のバージ...
<input checked="" type="checkbox"/> CUDA		
<input type="checkbox"/> NVIDIA GeForce Experience co...		
<input type="checkbox"/> Driver components		
<input type="checkbox"/> Other components		

ダウングレード版（バージョンの小さなもの）を追加でインストールする場合、「カスタムインストール」にて「CUDA」のみにチェックし、他はチェックを外す。

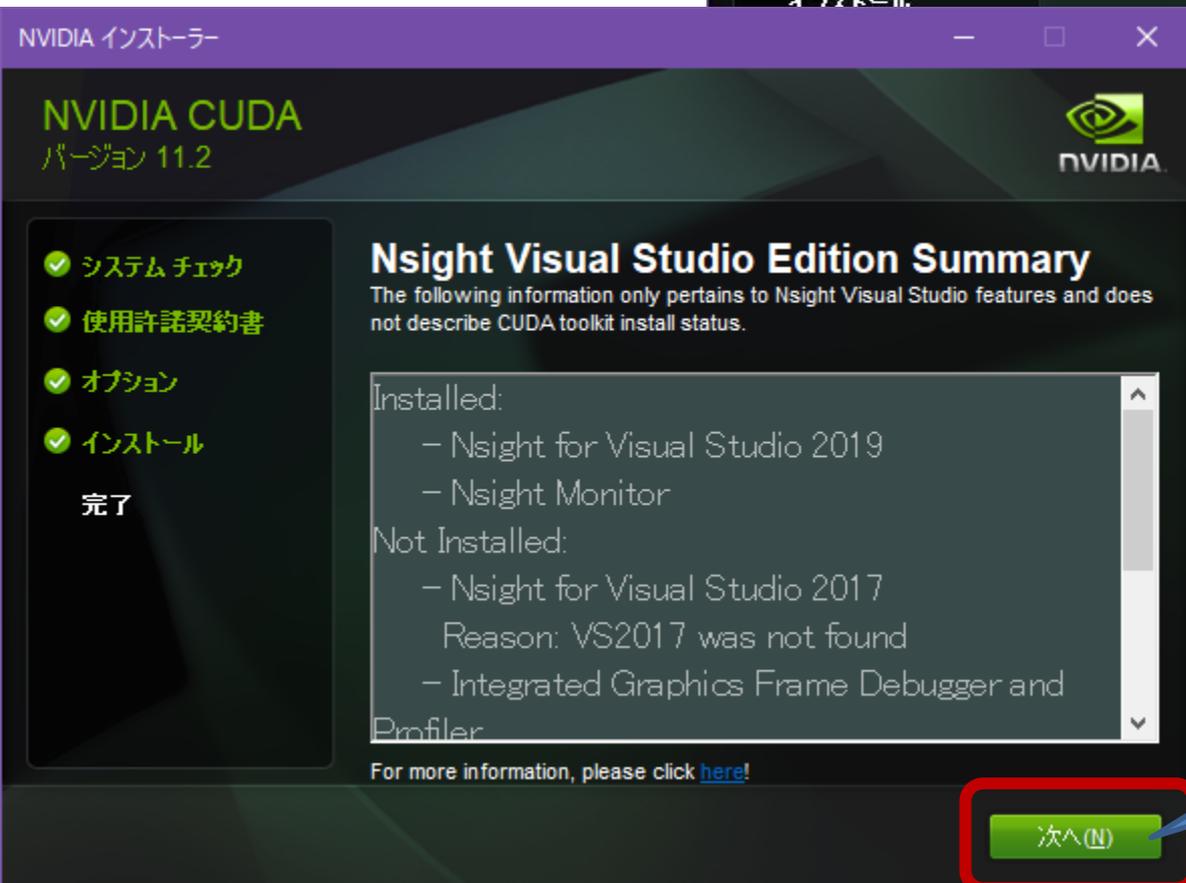
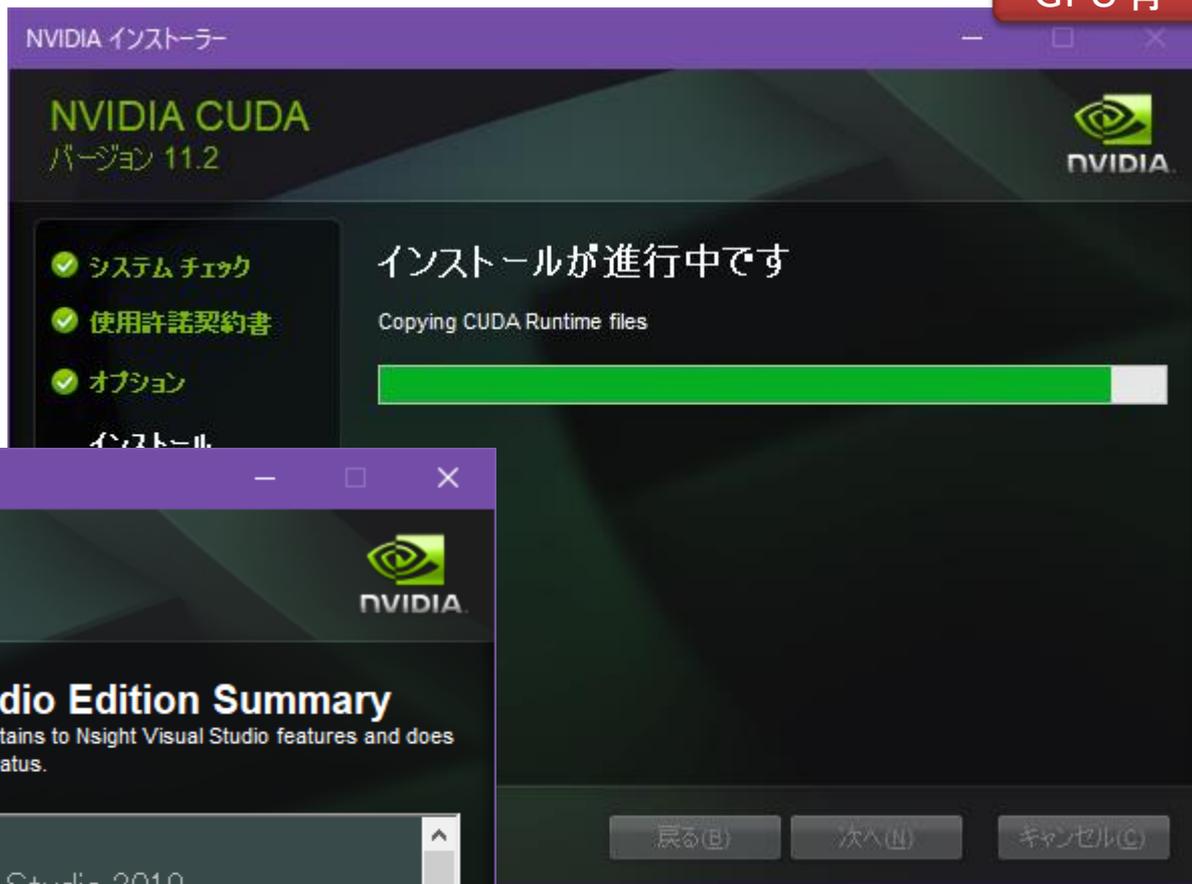
戻る(B) 次へ(N) キャンセル(C)



【次へ】をクリック...

# TensorFlowの場合

GPU有



【次へ】をクリックし、終了

# cuDNNをcuDNN Archiveから ダウンロードする

- <https://developer.nvidia.com/rdp/cudnn-archive>

## cuDNN Archive

Download cuDNN v8.7.0 (November 28th, 2022), for CUDA 11.x

Download cuDNN v8.7.0 (November 28th, 2022), for CUDA 10.2

Download cuDNN v8.6.0 (October 3rd, 2022), for CUDA 11.x

Download cuDNN v8.6.0 (October 3rd, 2022), for CUDA 10.2

Download cuDNN v8.5.0 (October 3rd, 2022), for CUDA 11.x

今回は、CUDA11用のcuDNN 8.6.0を利用する  
こちらも一つか二つ前のバージョンが良い(経験上)

[Download cuDNN v8.7.0 \(November 28th, 2022\), for CUDA 10.2](#)

[Download cuDNN v8.6.0 \(October 3rd, 2022\), for CUDA 11.x](#)

クリックすると下記が  
展開・表示される

## Local Installers for Windows and Linux, Ubuntu(x86\_64, armsbsa)

[Local Installer for Windows \(Zip\)](#)

Windows用を選択する

[Local Installer for Linux x86\\_64 \(Tar\)](#)

[Local Installer for Linux PPC \(Tar\)](#)

[Local Installer for Linux SBSA \(Tar\)](#)

[Local Installer for Ubuntu18.04 x86\\_64 \(Deb\)](#)

[Local Installer for Debian 11 \(Deb\)](#)

[Local Installer for Ubuntu20.04 x86\\_64 \(Deb\)](#)

[Local Installer for Ubuntu22.04 x86\\_64 \(Deb\)](#)

[Local Installer for Ubuntu20.04 aarch64sbsa \(Deb\)](#)

[Local Installer for Ubuntu20.04 cross-sbsa \(Deb\)](#)

## Local Installers for Red Hat (x86\_64,

armv7hl, Power architecture)

Home

## NVIDIA Developer Program Membership Required

The file or page you have requested requires membership in the NVIDIA Developer Program. Please either log in or join the program to access this material. [Learn more](#) about the benefits of the NVIDIA Developer Program.

Login Join now

**NVIDIA DEVELOPER**

- HIGH PERFORMANCE COMPUTING
- GAMEWORKS
- JETPACK
- DRIVE

cuDNNを利用する場合、NVIDIAに登録しなければならないので注意。

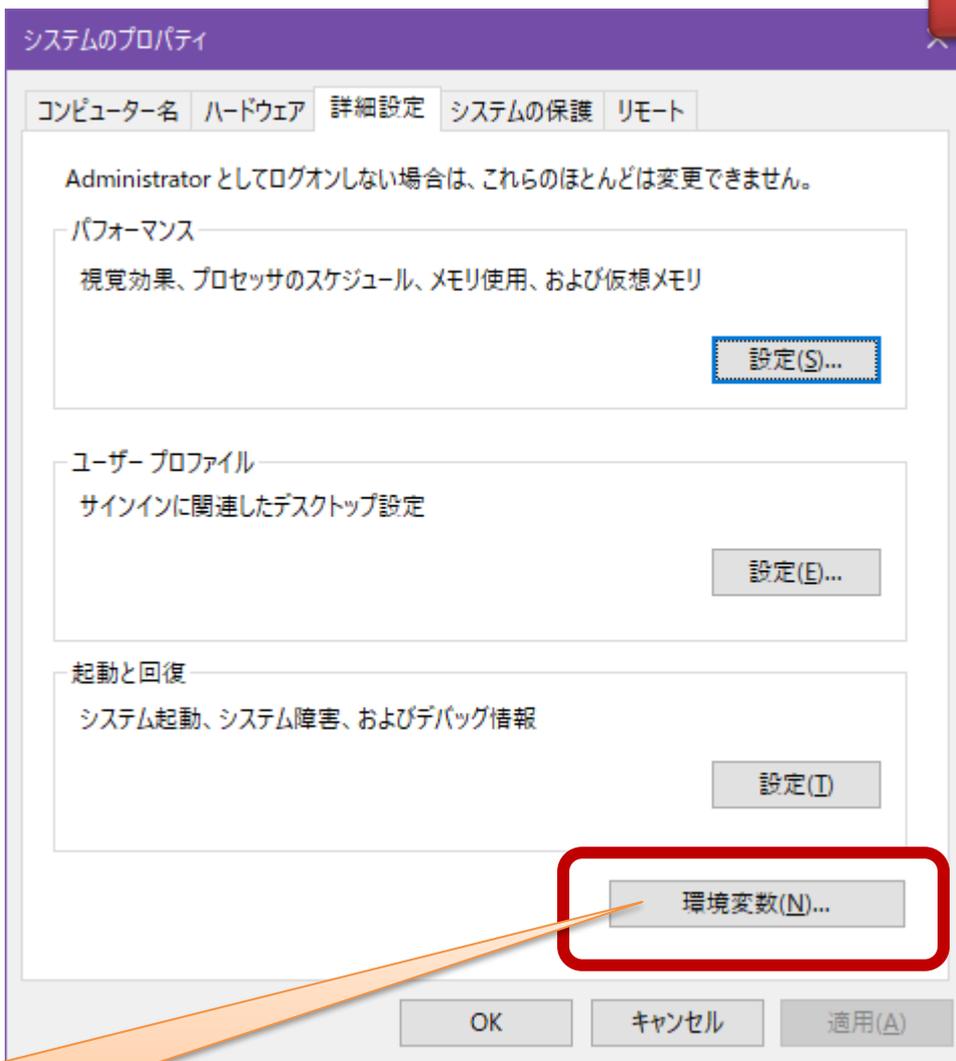
cuDNNはダウンロードした圧縮ファイルを解凍しておく。  
この後、環境変数の変更時に明らかになるCUDAのフォルダ内部に解凍されたファイルを移動(ないしコピー)して使うことになる。

## CUDAを利用するための環境変数の確認・設定

The image shows a Windows Start menu with the 'システム(Y)' (System) option highlighted by a red dashed box. An arrow points from this option to the Windows Settings application. In the Settings app, the 'システム' (System) category is selected, and the 'システムの詳細設定' (Advanced system settings) link is highlighted by a red dashed box. A callout box provides instructions on how to reach these settings.

環境変数を設定する

- ①【スタート】を右クリックし、【システム】を選択
- ②【システムの詳細設定】をクリック



環境変数... をクリック

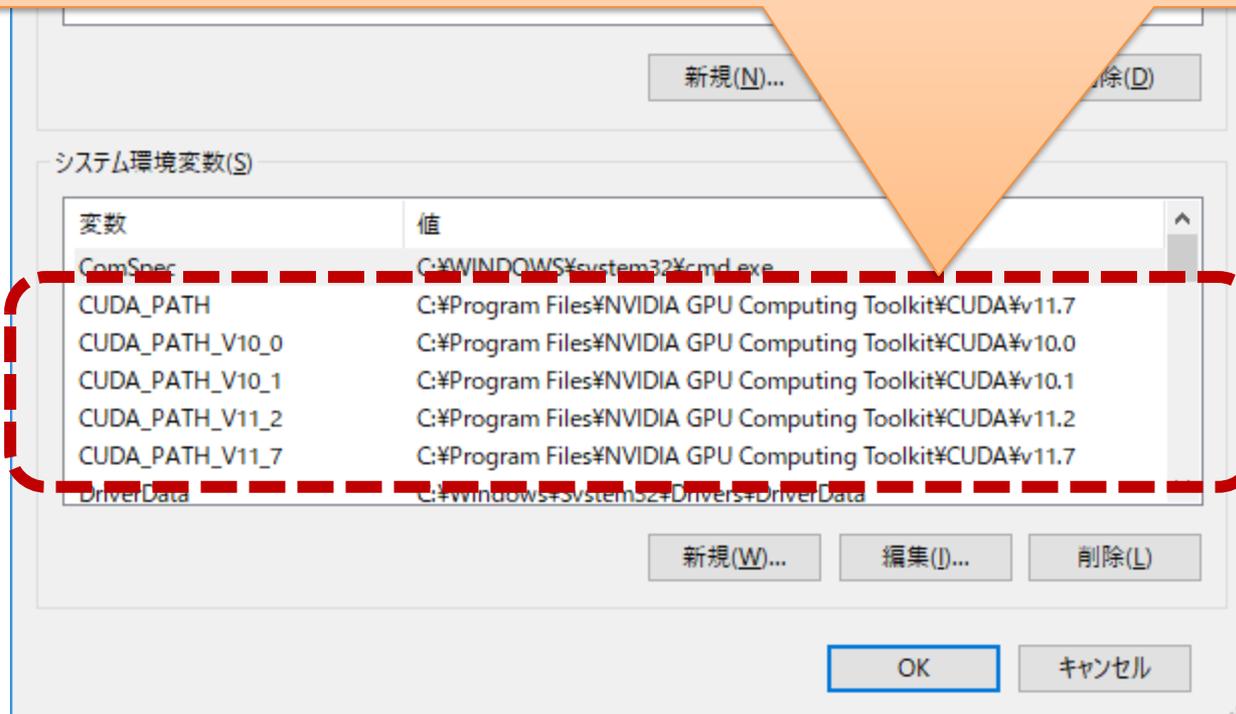
環境変数

複数のCUDAをインストールしている場合には、  
CUDA\_PATHを適切なもの書き換える。

※CUDA\_PATH\_V##から適切なものを選択し、同じ表記に変更する。

また、cuDNN内部のファイルをこれらのフォルダに移動(ないしコピー)する  
必要があるので、パスを控えておく。

今回はCUDA\_PATHをv11.7に設定



# TensorFlowの場合

GPU有

システム環境変数(S)

変数	値
NVCUDASAMPLES11_2_ROOT	C:\ProgramData\NVI...
NVTOOLSEXT_PATH	C:\Program Files\NVI...
OS	Windows_NT
Path	C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.7\bin;...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PIPENV_VENV_IN_PROJECT	true
POWERSHELL_DISTRIBUTIO...	MSI:Windows 10 Pro

【システム環境変数】の path を編集。

環境変数名の編集

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.7\bin

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.7\libn...

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.2\bin

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.2\libn...

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v10.1\bin

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v10.1\libn...

C:\Users\HN\AppData\Roaming\ActiveState\bin

C:\Program Files (x86)\Common Files\Oracle\Java\javapath

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v10.0\bin

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v10.0\libn...

C:\Program Files\Java\jdk1.8.0\_271\bin

C:\Program Files\Microsoft MPI\Bin\

C:\WINDOWS\system32

C:\WINDOWS

C:\WINDOWS\System32\Wbem

C:\WINDOWS\System32\WindowsPowerShell\v1.0\

C:\Program Files\dotnet\

C:\Program Files\Microsoft SQL Server\130\Tools\Binn\

C:\Users\HN\dnx\bin

C:\Program Files\Microsoft DNX\Dnvm\

新規(N)

編集(E)

参照(B)...

削除(D)

上^ (U)

下^ (D)

テキストの編集(T)...

OK

キャンセル

新規(W)...

編集(I)...

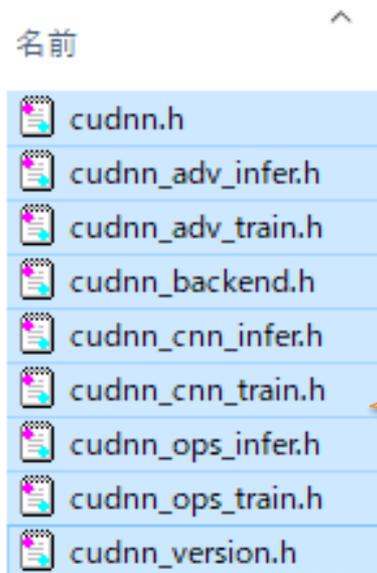
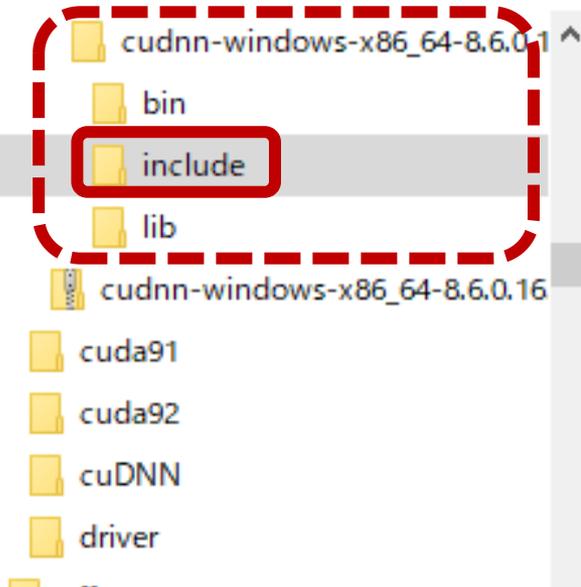
削除(L)

OK

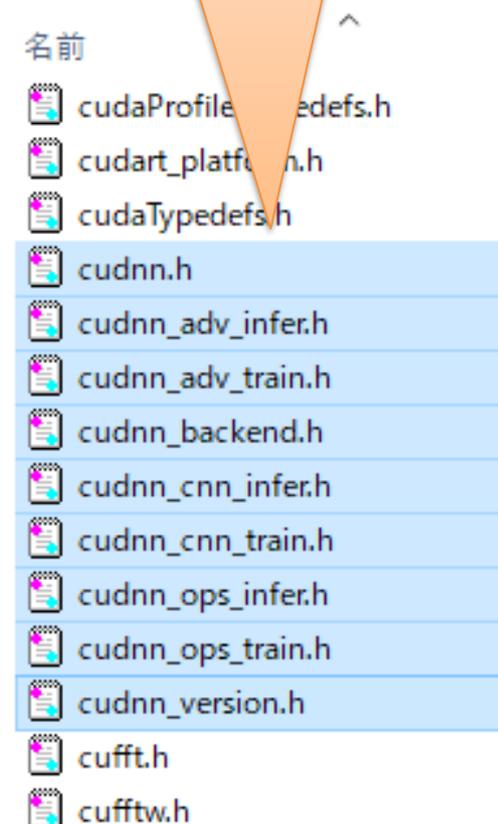
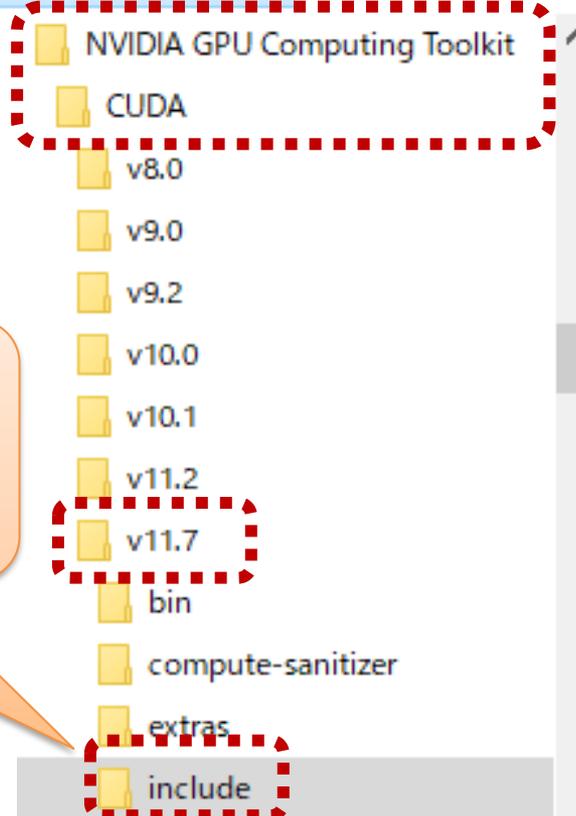
キャンセル

CUDAのbinおよびlibnvvpの2つのパスについて、適切なバージョンのものを上に移動させる。



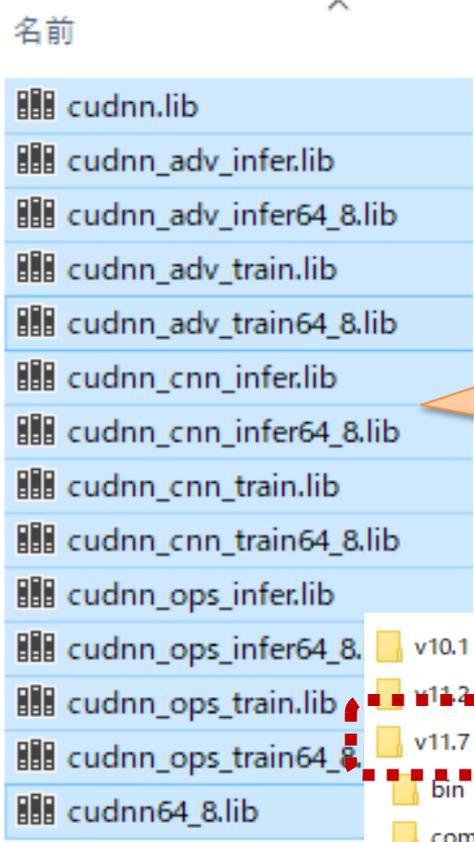
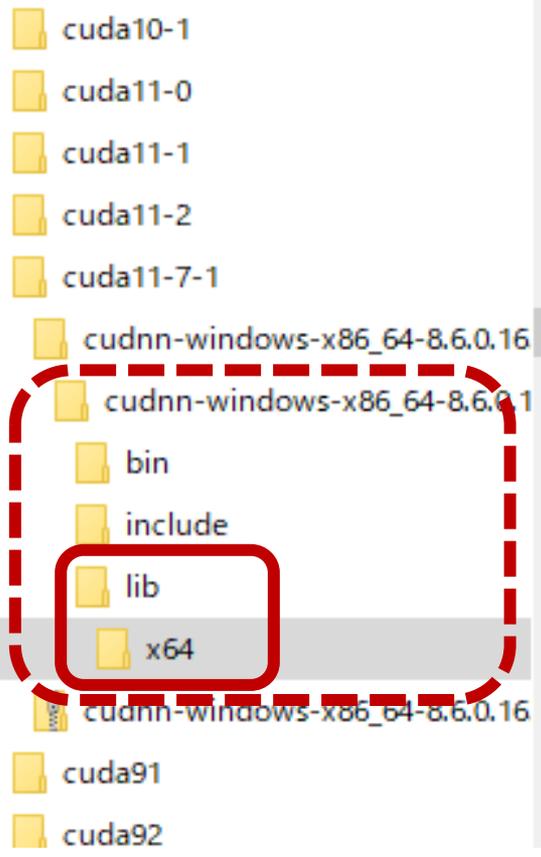


解凍したcuDNNの、include内のファイルを下記の場所にコピー(ないし移動)する。



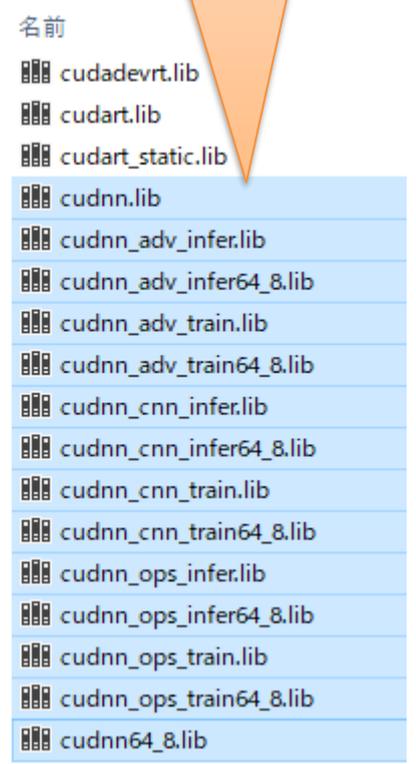
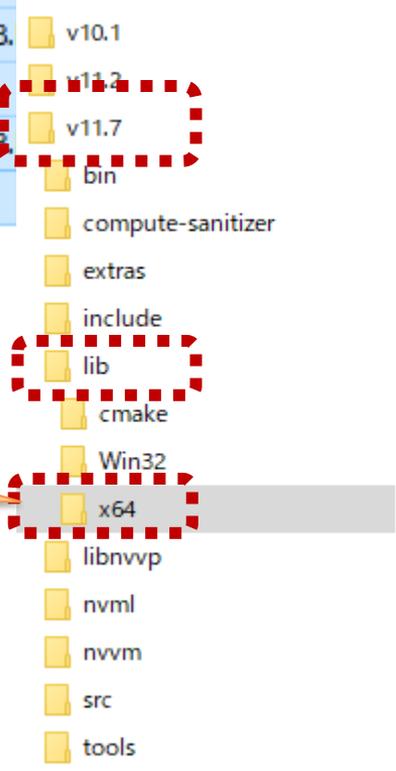
該当するバージョンのCUDAがインストールされた場所(システム環境変数にて控えた場所)内部のincludeフォルダ

# TensorFlowの場合



解凍したcuDNNの、lib → x64内のファイルを下記の場所にコピー(ないし移動)する。

該当するバージョンのCUDAがインストールされた場所(システム環境変数にて控えた場所)内部のlib → x64フォルダ



# CUDAおよびcuDNNのバージョン確認

- CUDAのバージョン確認

- `nvcc -V`

```
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005–2022 NVIDIA Corporation
Built on Wed_Jun__8_16:59:34_Pacific Daylight Time_2022
Cuda compilation tools, release 11.7, V11.7.99
Build cuda_11.7.r11.7/compiler.31442593_0
```

通常releaseの数字の前に「v」を付けたフォルダ内に下記cuDNN関連のファイルも入っている。

- cuDNNのバージョン確認

- `type "C:¥Program Files¥NVIDIA GPU Computing Toolkit¥CUDA¥v***¥include¥cudnn_version.h"`

- 「\*\*\*」の部分を上記のCUDAのバージョンに変え「#define」の部分にて確認。

```
#define CUDNN_MAJOR 8
#define CUDNN_MINOR 6
#define CUDNN_PATCHLEVEL 0
```

# NVIDIA-GPUをpipenv systemで使う (1).

- “Visual Studio 20xx の Microsoft Visual C++ 再頒布可能パッケージ” をダウンロードし、インストールする。
  - <https://visualstudio.microsoft.com/ja/vs/older-downloads/>
  - “Visual Studio Dev Essentials” への入会が必要
  - 「最新の Microsoft Visual C++ 再頒布可能パッケージバージョン」をダウンロードしてインストールする。

# Visual Studio サブスクライバーのダウンロード

古いバージョンでは、アクティブな Visual Studio Subscription が必要です。以下の製品を選択し、[ダウンロード] ボタンをクリックして Visual Studio (MSDN) サブスクリプションにログインし、古いバージョンにアクセスします。

[すべて展開 →](#) | [すべて折りたたむ →](#)

> 2019

> 2017

> 2015

> Visual Studio for Mac

> 分離シェルと統合シェル

> その他の Tools、Frameworks、そして Redistributables

✓ その他の Tools、Frameworks、そして Redistributables

Visual C++ 再頒布  
可能パッケージ

Visual C++ 再頒布可能パッケージでは、Visual Studio を使用して構築された C++ アプリケーションを実行するために必要なランタイム コンポーネントがインストールされます。

利用可能なバージョンにアクセスするには、[ダウンロード] をクリックします。

ダウンロード

Microsoft Build  
Tools 2015 Update 3

マネージ アプリケーションのビルドに必要なツールです。これらのツールは、以前の .NET Framework に含まれていましたが、このダウンロードから個別に入手できるようになりました。

ダウンロード

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

learn.microsoft.com/ja-jp/cpp/windows/latest-supported-vc-redist?view=msvc-170

Learn 発見 ▾ 製品ドキュメント ▾ 開発言語 ▾ トピック ▾

C++ Visual Studio での C++ の概要 言語リファレンス ▾ ライブラリ ▾ C++ビルド プロセス ▾ C++を使用した Windows プログラミング ▾

バージョン

Visual Studio 2022 ▾

🔍 タイトルでフィルター

Visual C++ アプリケーションの依存関係の理解

再配布する DLL の決定

展開方法を選択する

ユニバーサル CRT の配置

▽ Visual C++ ファイルの再配布

Visual C++ ファイルの再配布

マージ モジュールを使用したコンポーネントの再配布

Visual C++ ActiveX コントロールの再配布

MFC ライブラリの再配布

ATL アプリケーションの再配布

サポートされている最新の Visual C++ 再頒布可能パッケージのダウンロード

> 配置例

Learn / C++, C、およびアセンブラー /

🔍 🌐 🏠 ☰

## サポートされている最新の Visual C++ 再頒布可能パッケージのダウンロード

[アフィリエイト] • 2024/03/01 • 6 人の共同作成者

🗨️ フィードバック

### この記事の内容

Visual Studio 2015、2017、2019、および 2022

最新の Microsoft Visual C++ 再頒布可能パッケージバージョン

Visual Studio 2013 (VC++ 12.0)

Visual Studio 2012 (VC++ 11.0) Update 4 (サポートされなくなりました)

さらに 5 個を表示

Visual C++ 再頒布可能パッケージは、Microsoft C および C++ (MSVC) ランタイム ライブラリをインストールします。これらのライブラリは、Microsoft C および C++ ツールを使用してビルドされた多くのアプリケーションで必要になります。アプリでこれらのライブラリを使用する場合は、アプリをインストールする前に、Microsoft Visual C++ 再頒布可能パッケージをターゲット システムにインストールしておく必要があります。再頒布可能パッケージ アーキテクチャは、ア

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

learn.microsoft.com/ja-jp/cpp/windows/latest-supported-vc-redist?view=msvc-170#latest-microsoft-visual-c-redistributable-version

## バージョン

Visual Studio 2022

🔍 タイトルでフィルター

再配布する DLL の決定

展開方法を選択する

ユニバーサル CRT の配置

### Visual C++ ファイルの再配布

Visual C++ ファイルの再配布

マージ モジュールを使用したコンポーネントの再配布

Visual C++ ActiveX コントロールの再配布

MFC ライブラリの再配布

ATL アプリケーションの再配布

サポートされている最新の Visual C++ 再頒布可能パッケージのダウンロード

### 配置例

Web クライアント アプリケーションの再配布

Visual C++ アプリケーションの ClickOnce 配置

以前のランタイム バージョンでの C++ -clr アプリケーションの実行

COM および .NET 用の C++ の属性

属性プログラミングの FAQ

グループ別の属性

使用法別の属性

属性リファレンス (アルファベット順)

## 最新の Microsoft Visual C++ 再頒布可能パッケージバージョン

最新バージョンは **14.40.33810.0** です

サポートされているアーキテクチャごとにこのバージョンをダウンロードするには、次のリンクを使います。

🔗 テーブルを展開する

Architecture	リンク	メモ
ARM64	<a href="https://aka.ms/vs/17/release/vc_redist.arm64.exe">https://aka.ms/vs/17/release/vc_redist.arm64.exe</a>	サポートされている最新の ARM64 バージョンの固定リンク
X86	<a href="https://aka.ms/vs/17/release/vc_redist.x86.exe">https://aka.ms/vs/17/release/vc_redist.x86.exe</a>	サポートされている最新の x86 バージョンの固定リンク
X64	<a href="https://aka.ms/vs/17/release/vc_redist.x64.exe">https://aka.ms/vs/17/release/vc_redist.x64.exe</a>	サポートされている最新の x64 バージョンの固定リンク。X64 再頒布可能パッケージには、ARM64 と X64 の両方のバイナリが含まれています。このパッケージを使うと、X64 再頒布可能パッケージが ARM64 デバイスにインストールされている場合に、必要な Visual C++ ARM64 バイナリを簡単にインストールできます。

長期サービス リリース チャンネル (LTSC) バージョンを含む他のバージョンは、[my.visualstudio.com](https://my.visualstudio.com) からダウンロードします。

## メモ

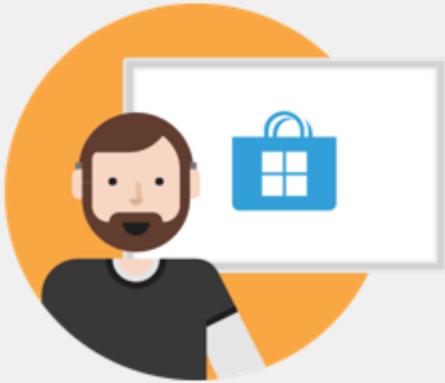
- Visual Studio 2015-2022 の Visual C++ 再頒布可能パッケージには、言語ごとに個別のパッケージはありません。これには、サポートされているすべての言語の EULA が含まれています。

Tensorflowに伴う  
GPU周りの設定はここまで

# Python3

- Pythonのサイト
  - <https://www.python.org/>
- Python3をダウンロードする
  - python2もあるが、古いコード体系なので注意





**インストールしようとしているアプリは、Microsoft 検証済みアプリではありません**

インストール対象を Microsoft Store のアプリのみに限定すると、PC の保護とスムーズな実行に役立ちます。

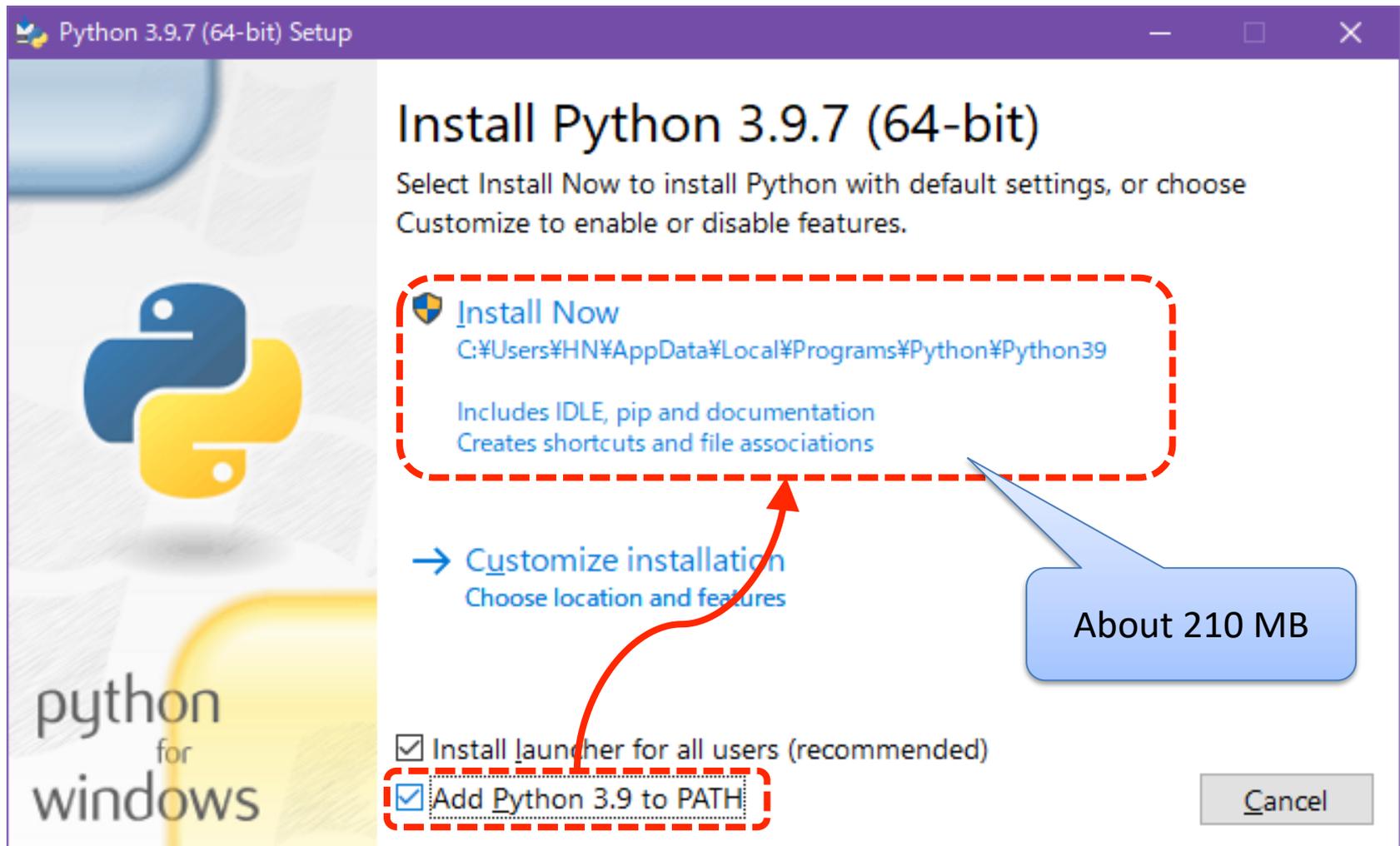
アプリを Microsoft Store から入手

インストールする

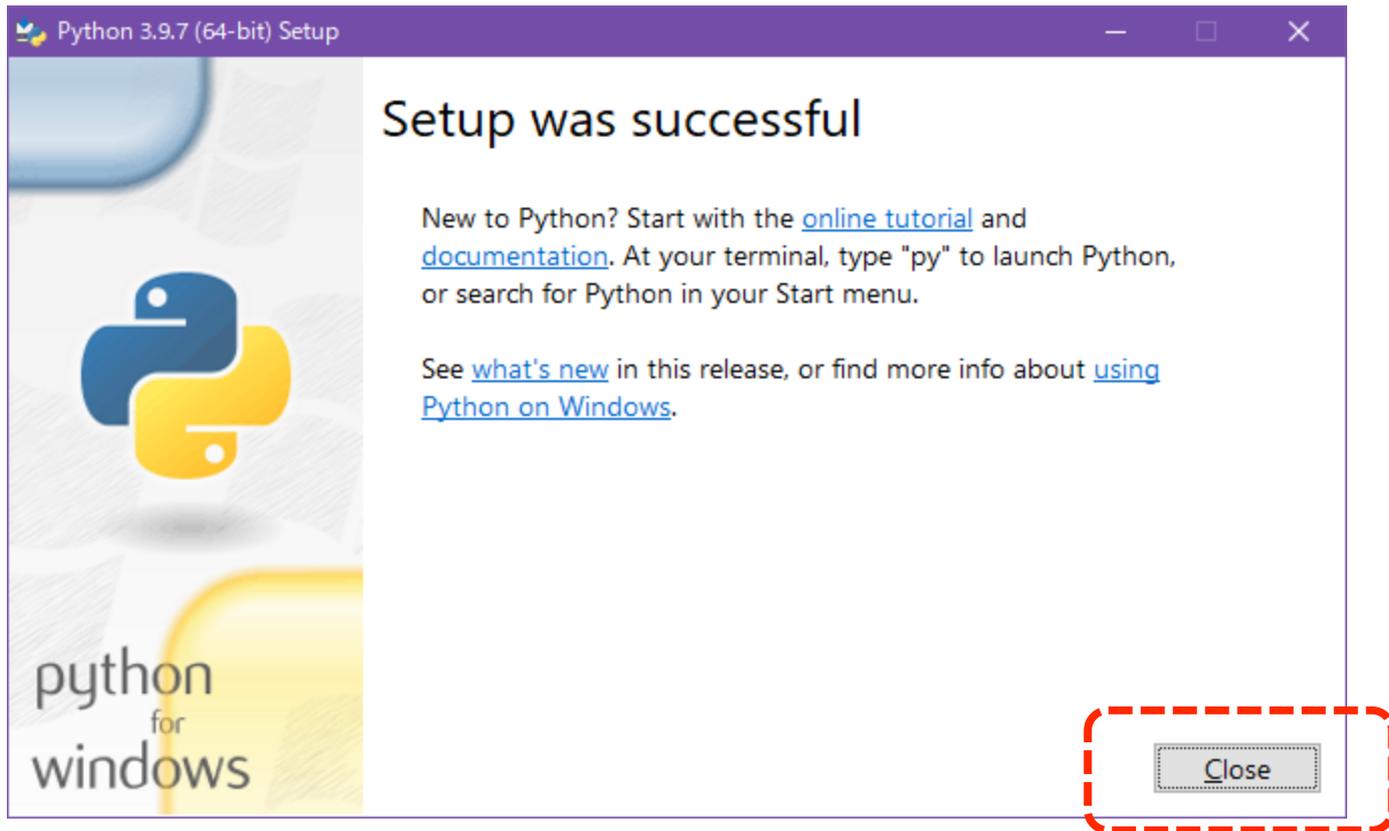
[アプリのおすすめの設定を変更 >](#)

# インストールの実施

“python-3.9.7-amd64.exe”の場合  
※過去にインストール済みで上位バージョンを  
インストールするか初めてインストールする場合



# インストール終了



# インストール状態の確認

- “cmd.exe” (コマンドプロンプト)を実行
- “python” を入力して実行
- 数行のメッセージ中に表示されるバージョンを確認し、“exit()” と入力して実行し、終了する。

```
E:¥> python
Python 3.9.7 (tags/v3.9.7:
Type "help", "copyright",
>>> exit()
E:¥>
```

End with “exit()”

- python
- >>> exit()

# Pythonの特定バージョンのインストールについて

バージョン番号  
で検索

The screenshot shows the Python website's search bar with '3.8.10' entered. A red dashed box highlights the search bar and the resulting page content. A red arrow points from the search bar to the page content. The page content includes the Python logo, a 'Donate' button, and a navigation menu with 'About', 'Downloads', 'Documentation', 'Community', 'Success Stories', 'News', and 'Events'. The main content area is titled 'Python 3.8.10' and contains the following text:

**Python 3.8.10**

**Release Date:** May 3, 2021

This is the tenth and final regular maintenance release of Python 3.8

**Note:** The release you're looking at is Python 3.8.10, a **bugfix release** for the legacy 3.8 series. *Python 3.11* is now the latest feature release series of Python 3. [Get the latest release of 3.11.x here.](#)

According to the release calendar specified in [PEP 569](#), Python 3.8.10 is the final regular maintenance release. Starting now, the 3.8 branch will only accept security fixes and releases of those will be made in source-only form until October 2024.

Compared to the 3.7 series, this last regular bugfix release is relatively dormant at 92 commits since 3.8.9. Version 3.7.8, the final regular bugfix release of Python 3.7, included 187 commits. But there's a bunch of important updates here regardless, the biggest being Big Sur and Apple Silicon build support. This work would not have been possible without the effort of Ronald Oussoren, Ned Deily, Maxime Bélangier, and Lawrence D'Anna from Apple. Thank you!

Take a look at the [change log](#) for details.

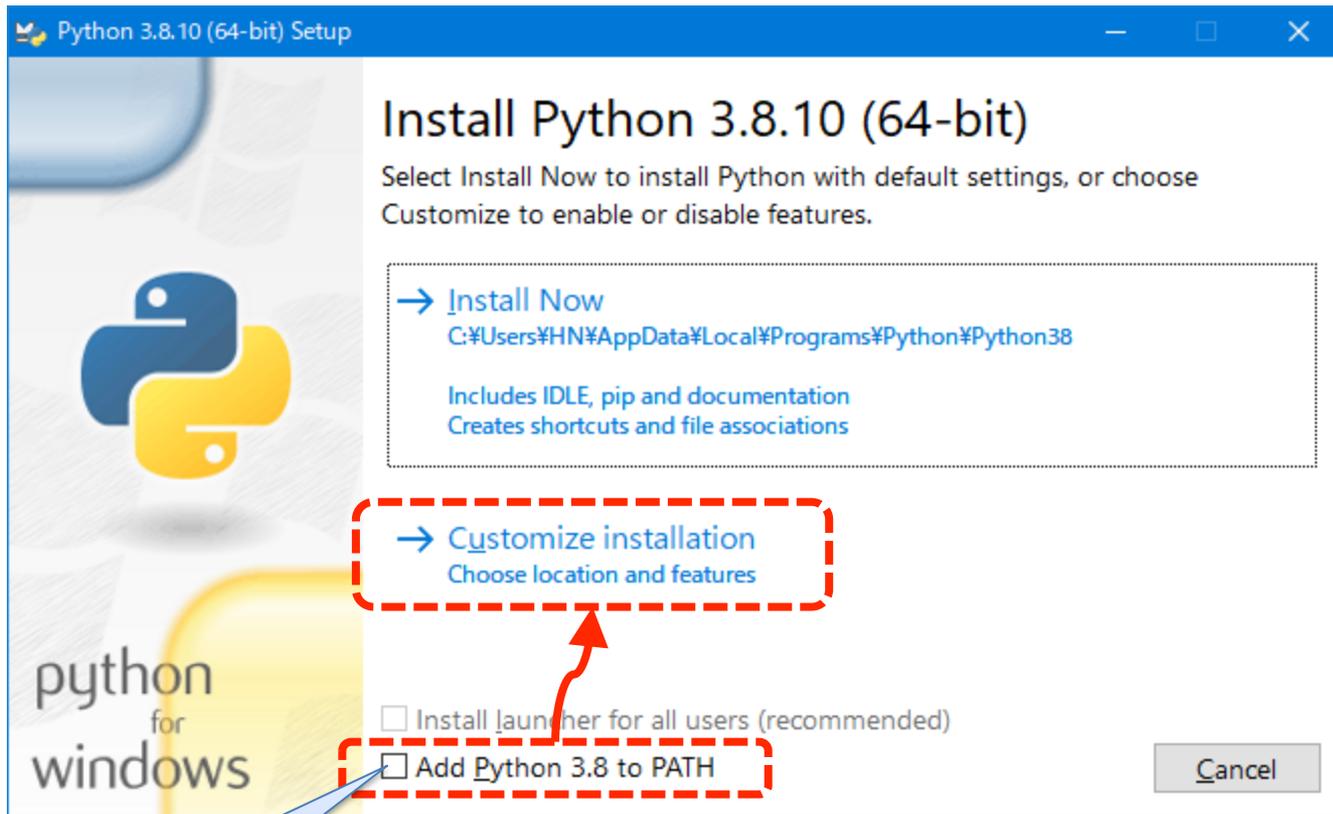
At the bottom of the page, there are two tabs: 'python-3.8.10-embed.exe' and 'python-3.8.10-embed.zip'. A '閉じる' (Close) button is visible in the bottom right corner.

# 該当バージョン情報の下方 「Files」内から該当ファイルをダウンロード

## Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
<a href="#">Gzipped source tarball</a>	Source release		83d71c304acab6c678e86e239b42fa7e	24720640	<a href="#">SIG</a>
<a href="#">XZ compressed source tarball</a>	Source release		d9eee4b20155553830a2025e4dcaa7b3	18433456	<a href="#">SIG</a>
<a href="#">macOS 64-bit Intel installer</a>	macOS	for macOS 10.9 and later	690ddb1be403a7efb202e93f3a994a49	29896827	<a href="#">SIG</a>
<a href="#">macOS 64-bit universal2 installer</a>	macOS	experimental, for macOS 11 Big Sur and later; recommended on Apple Silicon	ae8a1ae082074b260381c058d0336d05	37300939	<a href="#">SIG</a>
<a href="#">Windows embeddable package (32-bit)</a>	Windows		659adf421e90fba0f56a9631f79e70fb	7348969	<a href="#">SIG</a>
<a href="#">Windows embeddable package (64-bit)</a>	Windows		3acb1d7d9bde5a79f840167b166bb633	8211403	<a href="#">SIG</a>
<a href="#">Windows help file</a>	Windows		a06af1ff933a13f6901a75e59247cf95	8597086	<a href="#">SIG</a>
<a href="#">Windows installer (32-bit)</a>	Windows		b355cfc84b681ace8908ae50908e8761	27204536	<a href="#">SIG</a>
<a href="#">Windows installer (64-bit)</a>	Windows	Recommended	62cf1a12a5276b0259e8761d4cf4fe42	28296784	<a href="#">SIG</a>

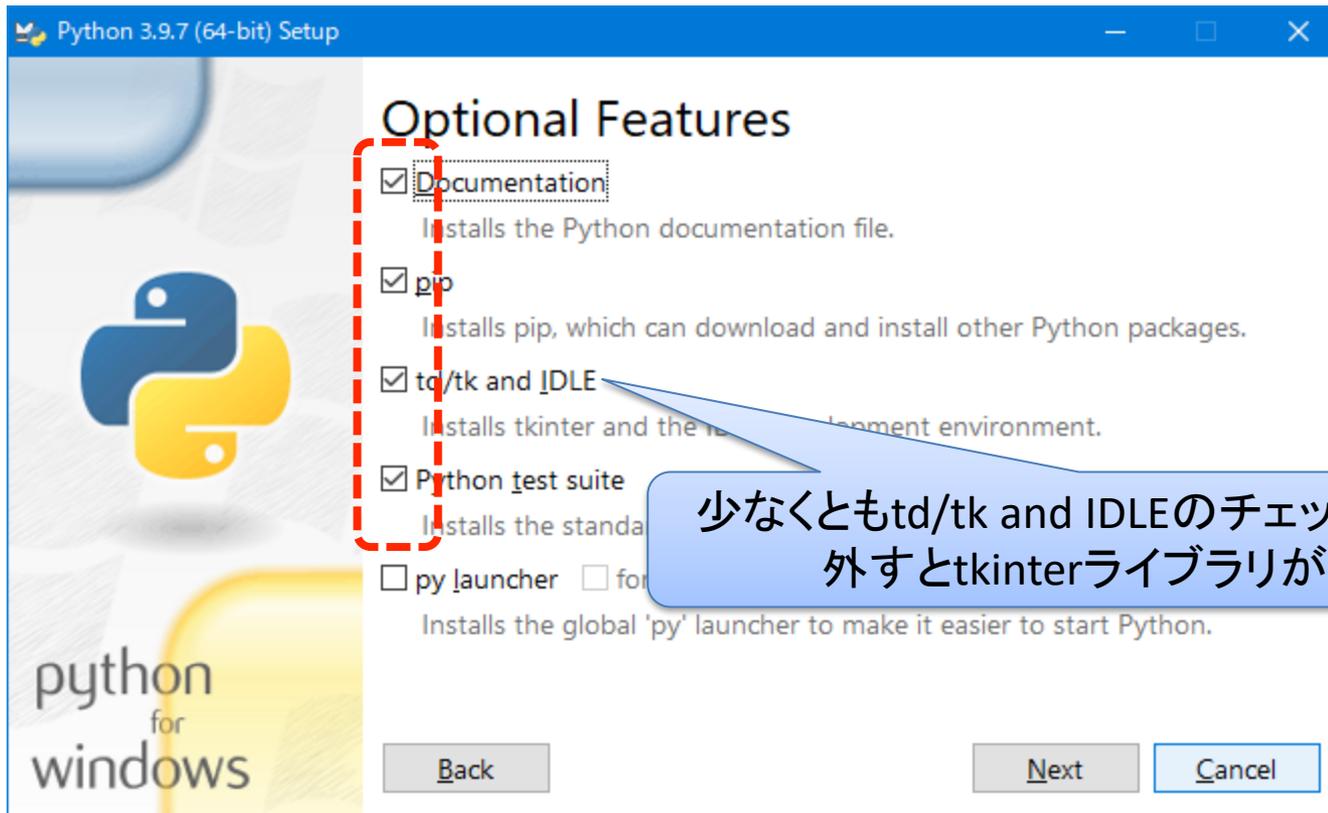
# 他のPythonのバージョンを追加でインストールする場合(1)



チェックを外す

Pipenvを使って複数のPythonを利用する場合、あらかじめ該当のPythonがインストールされている必要がある。  
※pipenv以外のvenv環境を利用する方法では不要な場合もある。

# 他のPythonのバージョンを追加でインストールする場合(2)

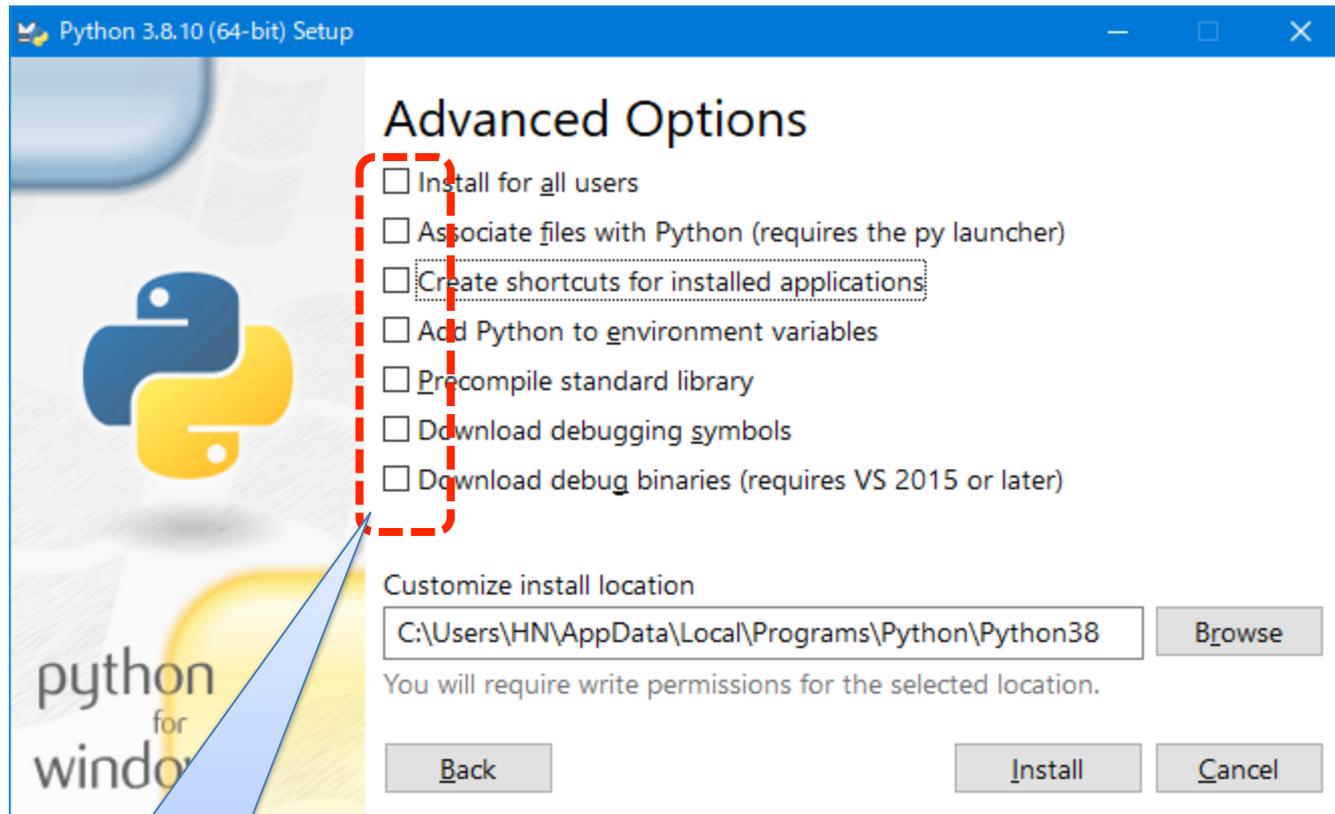


## 参考資料

<https://qiita.com/aujinen/items/b6f7ce3b727cd56bebb8>

Juliaにてpipenv仮想環境利用でPyPlot描画できず、結果的にpythonのtkinterインストールの罫に...

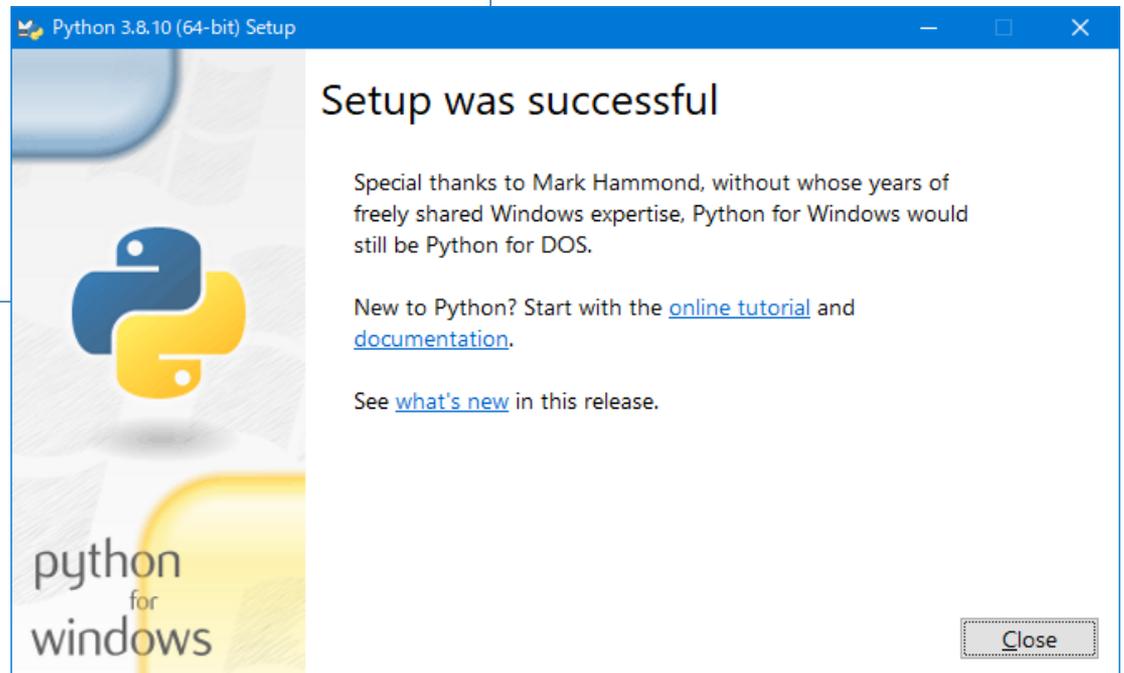
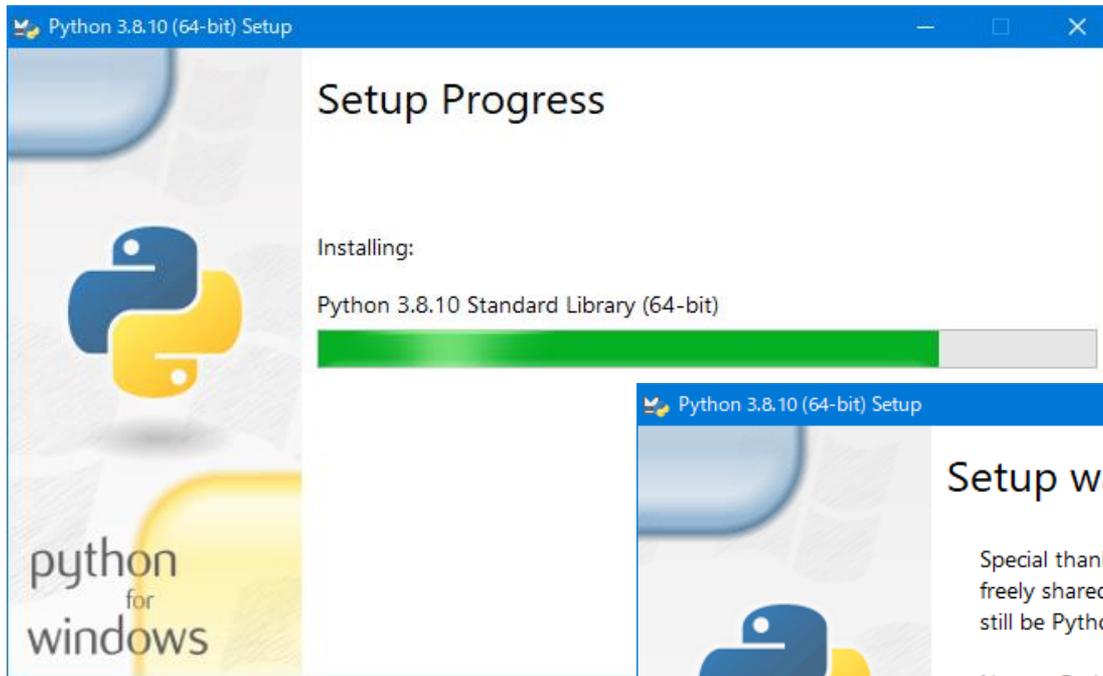
# 他のPythonのバージョンを追加でインストールする場合(3)



チェックを全て外す

現行のPythonからのアップグレードの場合には、add python to environment variablesにチェックを入れて、該当バージョンのPythonへのリンクを環境変数に埋め込んでおく。

# 他のPythonのバージョンを追加でインストールする場合(4)





# 初めてpipenvを使う場合

- pipenvをpipを使ってインストール

```
コマンド プロンプト
C:\Users\nisi> pip install pipenv
Collecting pipenv
  Using cached https://files.pythonhosted.org/9/pipenv-2018.11.26-py3-none-any.whl
Collecting virtualenv (from pipenv)
  Downloading https://files.pythonhosted.org/p/virtualenv-16.7.5-py2.py3-none-any.whl (3.3MB)
  100% |#####| 3.3MB
Requirement already satisfied: pip>=9.0.1 in c:\users\nisi\appdata\local\program\ages (from pipenv) (18.1)
Collecting certifi (from pipenv)
  Downloading https://files.pythonhosted.org/p/certifi-2019.9.11-py2.py3-none-any.whl (154kB)
  100% |#####| 154kB
Requirement already satisfied: setuptools>=36.2.1 in c:\users\nisi\appdata\local\program\ages (from pipenv) (40.6.2)
Collecting virtualenv-clone>=0.2.5 (from pipenv)
  Using cached https://files.pythonhosted.org/packages/ba/f8/50c2b7dbc99e05fce5e5b9d9a31f/virtualenv_clone-0.5.3-py2.py3-none-any.whl
Installing collected packages: virtualenv, certifi, virtualenv-clone, pipenv
Successfully installed certifi-2019.9.11 pipenv-2018.11.26 virtualenv-16.7.5 virtualenv-clone-0.5.3
You are using pip version 18.1, however version 19.2.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

➤ pip install pipenv

About 400 MB

# 既にpipenvを使っている場合 念のため、pipenvをアップデートする

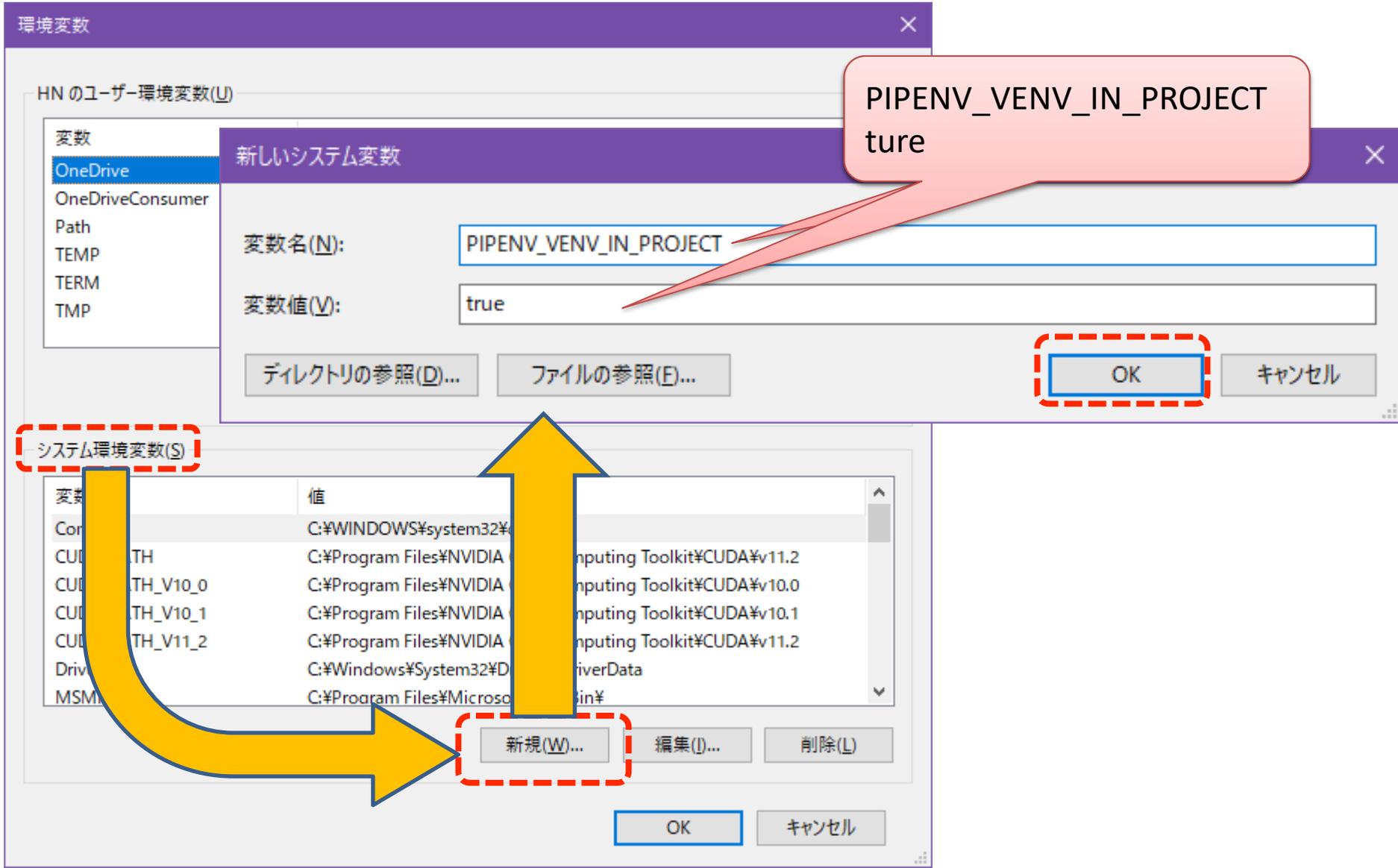
➤ pip install -U pipenv



```
Windows PowerShell  コマンドプロンプト - pip install -U
E:\dp\39TF26>pip install -U pipenv
Collecting pipenv
  Downloading pipenv-2021.5.29-py2.py3-none-any.whl (3.9 MB)
  |-----| 3.9 MB 3.3 MB/s
Collecting certifi
  Downloading certifi-2021.5.30-py2.py3-none-any.whl (145 kB)
  |-----| 145 kB 6.4 MB/s
Collecting virtualenv
  Downloading virtualenv-20.8.1-py2.py3-none-any.whl (5.3 MB)
  |-----| 5.3 MB 6.4 MB/s
Collecting virtualenv-clone>=0.2.5
  Downloading virtualenv_clone-0.5.7-py3-none-any.whl (6.6 kB)
Requirement already satisfied: pip>=18.0 in c:\users\%hn%\appdata\%
ipenv) (21.2.3)
Requirement already satisfied: setuptools>=36.2.1 in c:\users\%hn%\s
(from pipenv) (57.4.0)
Collecting filelock<4,>=3.0.0
  Downloading filelock-3.3.0-py3-none-any.whl (9.7 kB)
Collecting distlib<1,>=0.3.1
  Downloading distlib-0.3.3-py2.py3-none-any.whl (496 kB)
  |-----| 496 kB 3.3 MB/s
Collecting backports.entry-points-selectable>=1.0.4
```

# pipenvの環境変数設定

.venvという仮想用フォルダを作業領域内部に作る設定



## サブディレクトリを作成して移動し、“pipenv install”を実施

C:\ コマンドプロンプト

```
E:\> mkdir dp139TF26
```

```
E:\> cd dp139TF26
```

```
E:\dp139TF26>
```

- python 3.9.7をインストールするには、「pipenv --python 3.9」ないし「pipenv --python 3.9.7」を実行。

```
E:\dp139TF26> pipenv --python 3.9
```

```
Creating a virtualenv for this project...
```

```
Pipfile: E:\dp139TF26\Pipfile
```

```
Using C:/Users/HN/AppData/Local/Programs/Python/Python39/python.exe (3.9.7)
```

```
[====] Creating virtual environment...created virtual environment CPython3.9
```

```
creator CPython3Windows(dest=E:\dp139TF26\venv, clear=False, no_vcs_ignore
```

```
seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bu
```

```
Data\Local\pypa\virtualenv)
```

```
added seed packages: pip==21.2.4, setuptools==59.1.0, wheel==0.37.0
```

```
activators BashActivator, BatchActivator
```

```
Successfully created virtual environment
```

```
Virtualenv location: E:\dp139TF26\venv
```

```
Creating a Pipfile for this project.
```

➤ mkdir [sub]

➤ cd [sub]

➤ pipenv -- python 3.9

# TensorFlow-GPU関連の Pipenv仮想環境構築

※PyTorch関連のPipenv仮想環境構築は  
下記PDFを参照して下さい。

[https://www5.dent.niigata-  
u.ac.jp/~nisiyama/grad/Pipenv-PyTorch.pdf](https://www5.dent.niigata-u.ac.jp/~nisiyama/grad/Pipenv-PyTorch.pdf)

# GPU利用時

## tensorflow-gpuのインストール

- 例えば、2.6.0をインストールする場合
  - `pipenv install tensorflow-gpu==2.6.0`

```
E:¥dp139TF26 > pipenv install tensorflow-gpu==2.6.0
Installing tensorflow-gpu==2.6.0...
Adding tensorflow-gpu to Pipfile's [packages]...
Installation Succeeded
Pipfile.lock not found, creating...
Locking [dev-packages] dependencies...
Locking [packages] dependencies...
  Locking...Building requirements...
Resolving dependencies...
Success!
Updated Pipfile.lock (424499)!
Installing dependencies from Pipfile.lock (424499)...
===== 0/0 - 00:00:00
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
```

```
E:\dpl39TF26>pipenv run pip list
Package                               Version
-----
absl-py                               0.14.1
astunparse                            1.6.3
cachetools                            4.2.4
certifi                               2021.5.30
charset-normalizer                    2.0.6
clang                                  5.0
flatbuffers                           1.12
gast                                   0.4.0
google-auth                           1.35.0
google-auth-oauthlib                  0.4.6
google-pasta                           0.2.0
grpcio                                 1.41.0
h5py                                   3.1.0
idna                                   3.2
keras                                  2.6.0
Keras-Preprocessing                   1.1.2
Markdown                               3.3.4
numpy                                  1.19.5
oauthlib                               3.1.1
opt-einsum                             3.3.0
pip                                    21.2.4
protobuf                              3.18.0
pyasn1                                 0.4.8
pyasn1-modules                        0.2.8
requests                              2.26.0
requests-oauthlib                     1.3.0
rsa                                    4.7.2
setuptools                             58.1.0
six                                    1.15.0
tensorboard                           2.6.0
tensorboard-data-server                0.6.1
tensorboard-plugin-wit                 1.8.0
tensorflow-estimator                  2.6.0
tensorflow-gpu                         2.6.0
termcolor                              1.1.0
typing-extensions                      3.7.4.3
urllib3                                1.26.7
Werkzeug                               2.0.1
wheel                                  0.37.0
wrapr                                  1.12.1
```

tensorflow-gpuをインストール後の状況

➤ pipenv run pip list  
でリストアップしたもの

tensorflow-gpuをインストールすると自動的に  
h5py  
keras  
numpy  
がインストールされている。

# GPUを利用する場合の 必要なライブラリ 約1.5GB

- tensorflow-gpu
- numpy \*
- keras \*
- h5py \*

済

- jupyter
- matplotlib
- pandas

- pillow
- scikit-learn
- opencv-python
- scipy
- cython
- pydicom

※「tensorflow」をインストールすると、numpy, keras, h5pyが自動的にインストールされる。

「keras」のバージョンが「2.2.4」を超える場合は、古いPythonソースコードの「import keras」を「from tensorflow import keras」に変更し、また「from keras.XXX import XXX」を「from tensorflow.keras.XXX import」に変更する必要がある。

# “pipenv install [ライブラリ名]” コマンドでのライブラリのインストール

済 ➤ pipenv install tensorflow-gpu ==2.6.0

ライブラリの特定バージョンをインストールする方法

➤ pipenv install jupyter

➤ pipenv install matplotlib pandas

➤ ...

➤ pipenv install pydicom

jupyter以降、順に  
インストールする

複数のライブラリ名を1行に配置することにより、  
それらを同時にインストールできる。

エラーが発生した場合は、最後の2ページを参照してください。

# 例, “jupyter” ライブラリの “pipenv”環境へのインストール

C:\ コマンド プロンプト

```
pipenv install jupyter
```

```
Installing jupyter...
```

```
Adding jupyter to Pipfile's [packages]...
```

```
Installation Succeeded
```

```
Pipfile.lock (4918c7) out of date, updating to (ca72e7)...
```

```
Locking [dev-packages] dependencies...
```

```
Locking [packages] dependencies...
```

```
Success!
```

```
Updated Pipfile.lock (4918c7)!
```

```
Installing dependencies from Pi
```

```
=====
```

```
To activate this project's virtualenv, run pipenv shell.
```

```
Alternatively, run a command inside the virtualenv with pipenv run.
```

```
C:\Users\nisiy\dp136>
```

➤ pipenv install jupyter

# インストールし終えた ライブラリの一覧確認方法

```
コマンドプロンプト - pipenv shell
C:¥Users¥HN¥dp136¥pipenv shell
Launching subshell in virtual environment...
Microsoft Windows [Version 10.0.18362.418]
(c) 2019 Microsoft Corporation. All rights reserved.

(dp136-n6MCN6aI) C:¥Users¥HN¥dp136¥>pip list
Package            Version
-----
-eras              2.3.0
abs1-py            0.8.0
astor              0.8.0
attrs             19.1.0
backcall          0.1.0
bleach            3.1.0
colorama          0.4.1
cyclor            0.10.0
Other
```

GPU有

```
tensorflow-gpu    1.12.2
tensorflow         1.12.0
termcolor         1.1.0
terminado         0.8.2
testpath         0.4.2
tornado          6.0.3
traitlets        4.3.2
wcwidth          0.1.7
```

GPU無

```
tensorboard      1.14.0
tensorflow        1.14.0
tensorflow-estimator 1.14.0
termcolor        1.1.0
terminado        0.8.2
testpath         0.4.2
tornado          6.0.3
traitlets        4.3.2
```

```
(dp136-wo-gpu-66WSIfDe) C:¥Users¥HN¥dp136-wo-gpu¥>exit
```

```
C:¥Users¥HN¥dp136-wo-gpu¥
```

該当仮想環境を構築したディレクトリに入っている状態で

➤ pipenv shell  
にて、仮想環境を起動し

➤ pip list

を実行し、表示されるパッケージのリストを確認する。(インストールした以上のリストが出力されるので注意。)

確認後は、

➤ exit

コマンドで仮想環境から抜ける。

※仮想環境起動中はプロンプト行頭に(仮想環境名)が付随する。

# 【注意】protobufライブラリ関連

- protobufライブラリのバージョンが新しいとtensorflowライブラリを読み込んだ直後に下記のエラーが出ることがあります。プロトコルバッファが大幅更新された影響であり、互換性重視で稼働させる場合protobufを3.19.6に固定する必要があります。

```
In [1]: from tensorflow.python.client import device_lib
device_lib.list_local_devices()
```

```
556         default_value, message_type, enum_type, containing_type,
557         is_extension, extension_scope, options=None,
558         serialized_options=None,
559         has_default_value=True, containing_oneof=None, json_name=None,
560         file=None, create_key=None): # pylint: disable=redefined-builtin
--> 561     message.Message._CheckCalledFromGeneratedFile()
562     if is_extension:
563         return _message.default_pool.FindExtensionByName(full_name)
```

**TypeError:** Descriptors cannot not be created directly.

If this call came from a `_pb2.py` file, your generated code is out of date and must be regenerated with `protoc >= 3.19.0`.

If you cannot immediately regenerate your protos, some other possible workarounds are:

1. Downgrade the protobuf package to 3.20.x or lower.
2. Set `PROTOCOL_BUFFERS_PYTHON_IMPLEMENTATION=python` (but this will use pure-Python parsing and will be much slower).

More information: <https://developers.google.com/protocol-buffers/docs/news/2022-05-06#python-updates>

“pipenv run jupyter notebook”  
で実行する。

```
C:\Users\HN\dp\36> pipenv run jupyter notebook
[I 20:55:00.423 NotebookApp] Serving notebooks from local directory:
[I 20:55:00.424 NotebookApp] The Jupyter Notebook is running at:
```

Home Page - Select or create a n x +

localhost:8888/tree

jupyter

Files Running Clusters

Select items to perform actions on them.

	Name ↓	Last Modified	File size
<input type="checkbox"/>	/		
<input type="checkbox"/>	1_Classification	1ヶ月前	
<input type="checkbox"/>	Tkinter-test.ipynb	1ヶ月前	3.74 kB
<input type="checkbox"/>	Pipfile	1ヶ月前	324 B
<input type="checkbox"/>	Pipfile.lock	6分前	49.7 kB
<input type="checkbox"/>	pipList.txt	1ヶ月前	2.31 kB

▶ pipenv run jupyter notebook

# TensorFlow-GPU利用時注意点(1)

- Tensorflow-GPU 1.12.0にて、バージョン「2.2.4」の「keras」をインストールできず、「2.2.4」を超える場合は、「keras in tensorflow」(「2.1.6-tf」のバージョン)を使用する必要がある。
- この場合、Pythonソースコードの下記の変更が必要(jupyter notebook内部の置換コマンドで一斉に変更可能)
  - 変換前) `import keras`
    - 変換後) `from tensorflow import keras`
  - 変換前) `from keras.XXX import YYY`
    - 変換後) `from tensorflow.keras.XXX import YYY`

# TensorFlow-GPU利用時注意点(2)

- 特殊な変更が必要となるもの
  - 変換前本文中) `keras.utils.np_utils.XXX`
  - 変換後の本文中) `np_utils.XXX`
  - 変換後のライブラリ読み込み(冒頭)に追加
    - `from tensorflow.python.keras.utils import np_utils`

# 裏技・特定のPipenvの仮想環境を一発でインストールする

- 一度インストール終了したPipenvの仮想環境については、pipfileに設定内容が入っている。
- 設定済みのpipfileを新規の別フォルダにコピーし、同フォルダにて「pipenv install」とすれば、同一の仮想環境が構築される。
- 下記に、「Python=3.10、tensorflow-gpu=2.10.0、protobuf=3.19.6」にて設定したpipfileを置いておきます。
- [https://github.com/aujinen/colab\\_tfbook](https://github.com/aujinen/colab_tfbook)

# 【注意】TensorFlow-GPU

- WindowsでのTensorFlow-GPUは2.10まで
- 2.11以降のバージョンは未対応。
- Windowsで2.11以降を使いたい場合には、WSL2にて使うこと。

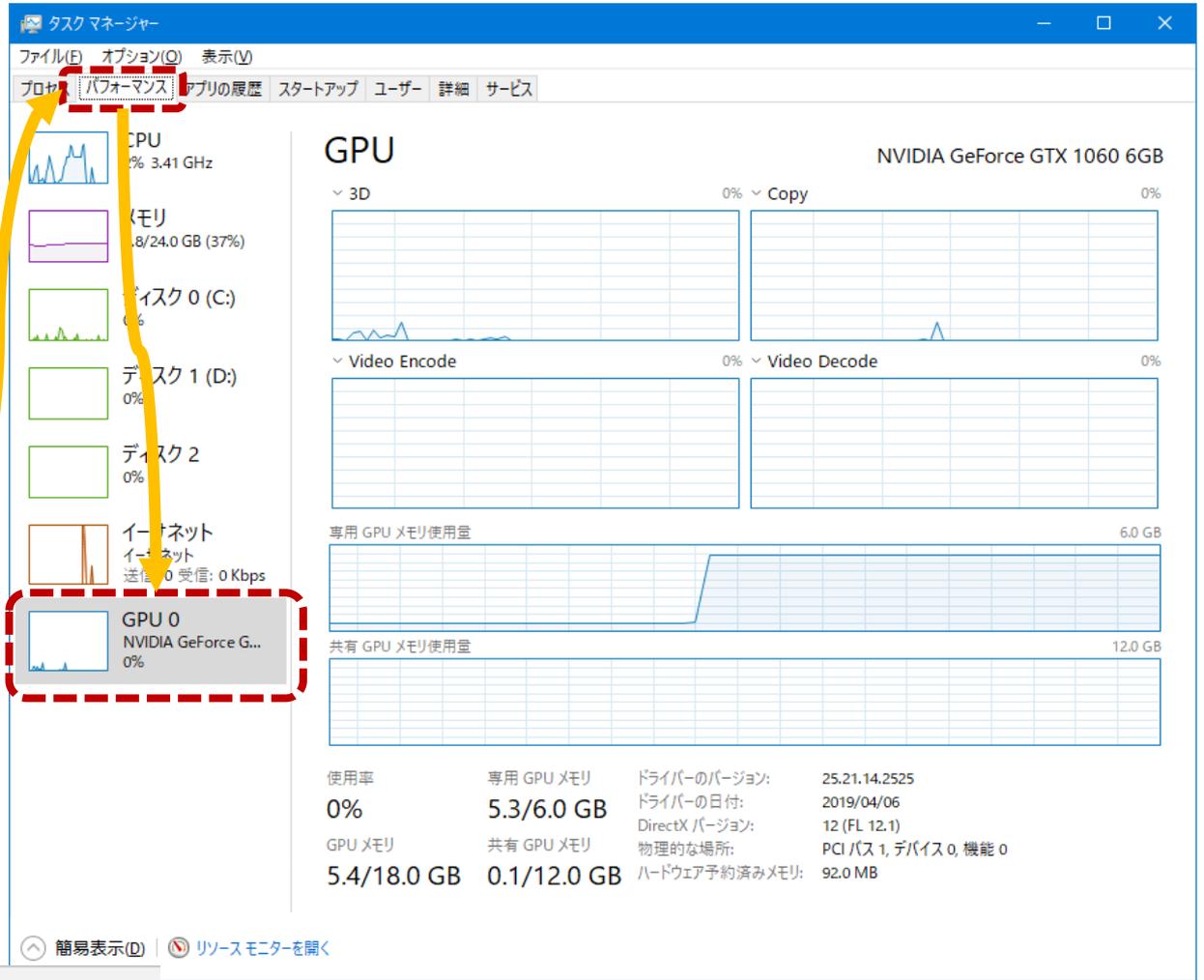
[https://www.tensorflow.org/install/source\\_windows](https://www.tensorflow.org/install/source_windows)

Install GPU support (optional)

See the Windows [GPU support](#) guide to install the drivers and additional software required to run TensorFlow on a GPU.

★ **Note:** GPU support on native-Windows is only available for 2.10 or earlier versions, starting in TF 2.11, CUDA build is not supported for Windows. For using TensorFlow GPU on Windows, you will need to build/install TensorFlow in WSL2 or use tensorflow-cpu with TensorFlow-DirectML-Plugin

# タスクマネージャーでのGPU稼働状況の確認



# Deep Learningの世界へ続く・・・

(次の2ページはインストールエラー用)

もし、インストール時に下記のようなエラーが出た場合、  
ライブラリをアンインストールして、再度インストールしなおす...

```
C:\> コマンド プロンプト
C:\Users\nisiy\dp\136> pipenv install opencv-python
Installing opencv-python...
Adding opencv-python to Pipfile's [packages]...
Installation Succeeded
Pipfile.lock (1cd370) out of date
Locking [dev-packages] dependencies...
Locking [packages] dependencies...
Locking Failed!
Traceback (most recent call last):
  File "c:\users\nisiy\appdata\local\pipenv\patched\pip\task\install.py", line 336, in _error_reraise
    raise ReadTimeoutError(self._opener, url, "Read timed out.")
pipenv.patched.pip._vendor.requests.exceptions.ReadTimeoutError: Read timed out.

... and retry...
```

➤ pipenv install [library]

... もしインストール時にエラー  
が出た場合

➤ pipenv uninstall [library]

... and retry...

```
C:\> コマンド プロンプト
C:\Users\nisiy\dp\136> pipenv uninstall opencv-python
Uninstalling opencv-python...
Removing opencv-python from Pipfile...
Locking [dev-packages] dependencies...
Locking [packages] dependencies...
```

インストールエラーが発生した場合は、ライブラリを削除して再試行してください。  
それでもうまくいかない場合は、次のページを参照してください。

エラー表示の最終行をチェックし、例えば「ReadTimeoutError」であれば、下記のようなサイトを検索して対処

- 英語版

- <https://github.com/pypa/pipenv/issues/3110>

- <https://github.com/pypa/pipenv/issues/1274>

- 日本語版

- [http://pynote.hatenablog.com/entry/python\\_ubuntu\\_pipenv#pexpectexceptionsTIMEOUT-%E3%82%A8%E3%83%A9%E3%83%BC](http://pynote.hatenablog.com/entry/python_ubuntu_pipenv#pexpectexceptionsTIMEOUT-%E3%82%A8%E3%83%A9%E3%83%BC)

- <https://qiita.com/kazetof/items/b18b225b7a3143514485>

# Pipenv VS-code関連資料の一部

- pipでアップデートするときのコマンド pip update
  - <https://qiita.com/HyunwookPark/items/242a8ceea656416b6da8>
- Windowsマシン上でVisual Studio Codeとpipenvを使ってPythonの仮想実行環境を構築する方法 (Jupyter notebookも)
  - <https://qiita.com/d-kitamura/items/3b8c4ac362668cac6e75>
- Pipenvことはじめ
  - <https://qiita.com/shinshin86/items/e11c1124e3e2e74556b8>

# 仮想環境でのVS-Code起動方法

- 仮想環境構築したフォルダで
  - code .
  - を実行する。

# Jupyter notebookの終了方法

## ※新しいバージョン

