# DLC Tool Lecture
# TensorFlow

2018. 02. 13.
Deep Learning College

모두의연구소 Research Scientist
**이일구**

# Contents

- TensorFlow 설치
- TensorFlow
- Graphs and Sessions

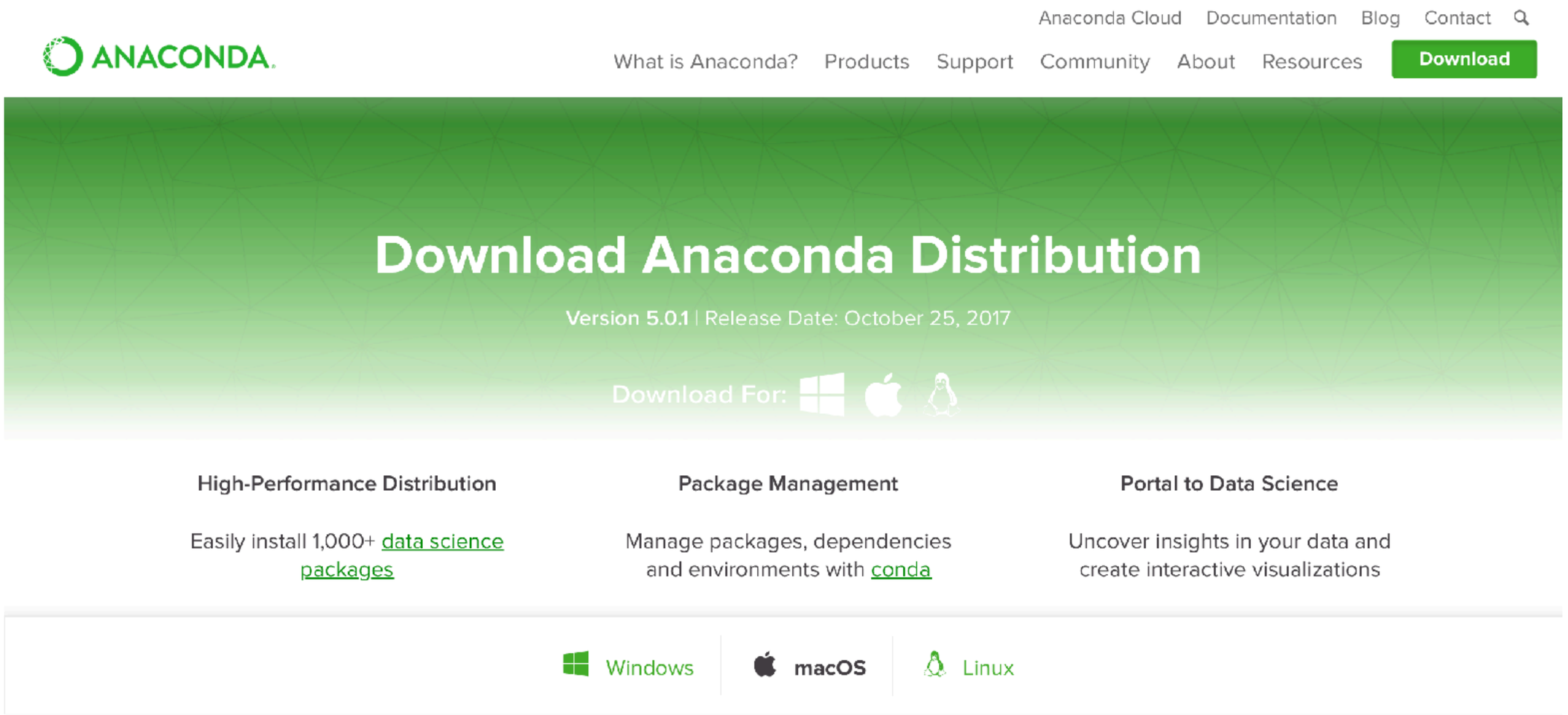
# TensorFlow 설치

# 기본 환경 설정

- Ubuntu 16.04 / MacOS
- Python 3.6 이상 (아나콘다 설치 추천)
- GPU 셋팅
  - CUDA® Toolkit 8.0, cuDNN v6 이상 (tf.__version__ < 1.5)
  - CUDA® Toolkit 9.0, cuDNN v7 이상 (tf.__version__ >= 1.5)
- TensorFlow 설치

# 아나콘다 설치

**https://www.anaconda.com/download**



# 아나콘다 설치

- 파일을 다운 받은 후 아래의 명령어를 실행한다

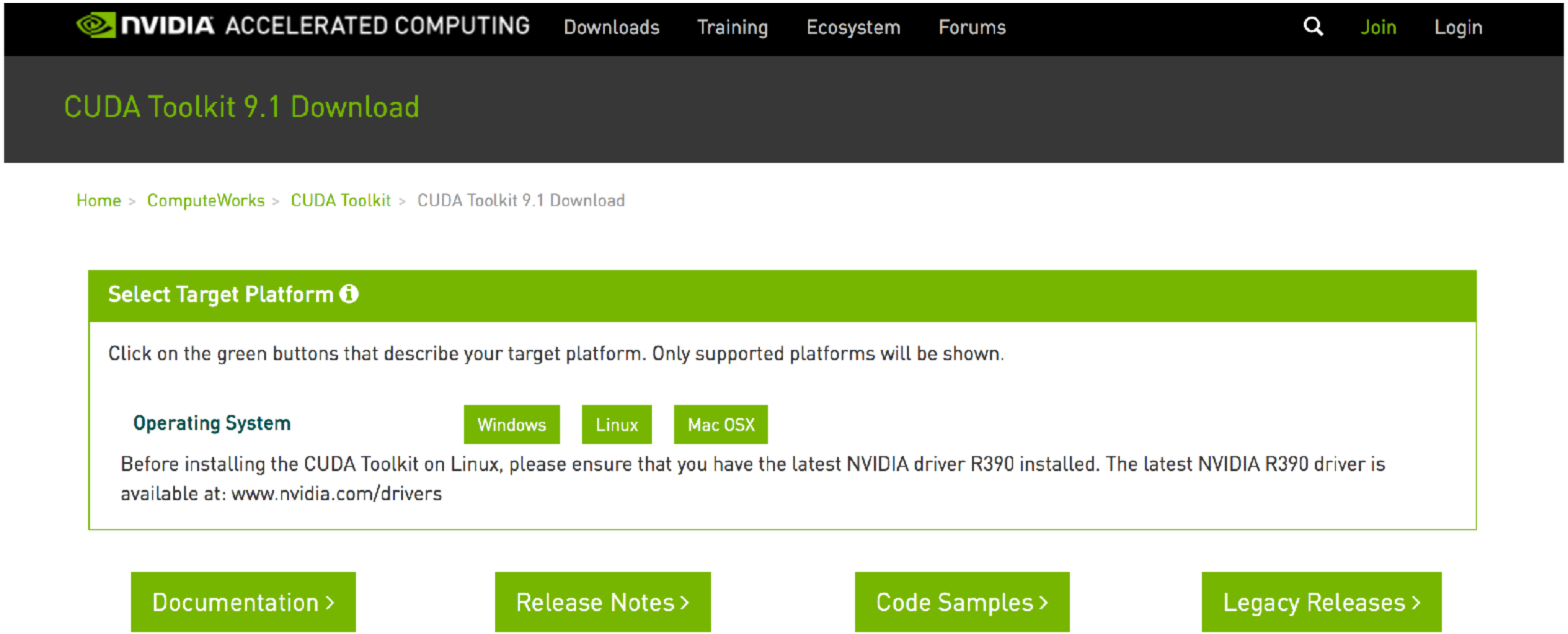
```
$ bash ~/Downloads/Anaconda3-5.0.1-Linux-x86_64.sh
```

# GPU 셋팅

- TensorFlow를 GPU에서 계산하고 싶으면
  - CUDA® Toolkit과 cuDNN library를 깔아야 한다
- if tf.__version__ < 1.5
  - CUDA® Toolkit 8.0 and cuDNN v6 library
- else
  - CUDA® Toolkit 9.1 and cuDNN v7 library

# CUDA® Toolkit 9.1

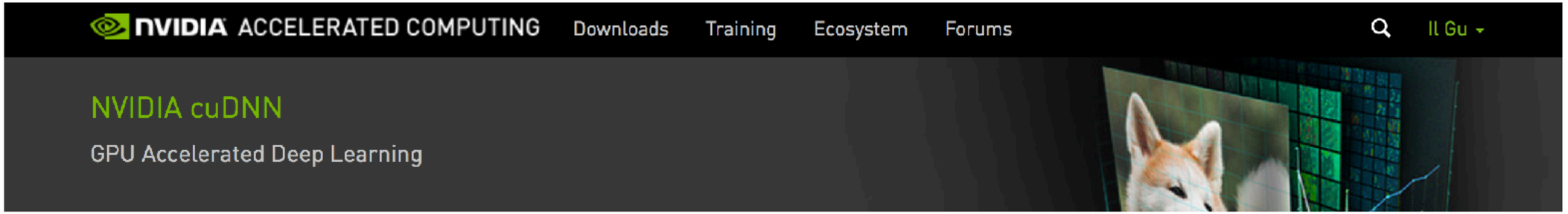
**https://developer.nvidia.com/cuda-downloads**



# cuDNN 7

회원가입 및 로그인 필요

**https://developer.nvidia.com/cudnn**




Home > ComputeWorks > Deep Learning > Software > NVIDIA cuDNN

The NVIDIA CUDA® Deep Neural Network library (cuDNN) is a GPU-accelerated library of primitives for deep neural networks. cuDNN provides highly tuned implementations for standard routines such as forward and backward convolution, pooling, normalization, and activation layers. cuDNN is part of the NVIDIA Deep Learning SDK.

Deep learning researchers and framework developers worldwide rely on cuDNN for high-performance GPU acceleration. It allows them to focus on training neural networks and developing software applications rather than spending time on low-level GPU performance tuning. cuDNN accelerates widely used deep learning frameworks, including Caffe2, MATLAB, Microsoft Cognitive Toolkit, TensorFlow, Theano, and PyTorch. See supported frameworks for more details. cuDNN is freely available to members of the NVIDIA Developer Program

## What's New in cuDNN 7?

**QUICKLINKS**

- Accelerated Computing - Training
- CUDA GPUs
- Tools & Ecosystem
- OpenACC: More Science Less Programming
- CUDA FAQ

# CUDA® Toolkit 9.1 설치

- 파일을 다운 받은 후 아래의 명령어를 실행한다
- Ctrl + Alt + F1을 눌러 콘솔 모드로 전환한다.
- 로그인 후 다음 명령어를 실행한다.

```
$ sudo service lightdm stop
$ chmod +x ./cuda_9.1.85_387.26_linux.run
$ sudo ./cuda_9.1.85_387.26_linux.run
$ sudo reboot
```

# CUDA Configure

- $HOME 폴더에 있는 .bashrc 파일을 직접 수정한다.

```
$ vi .bashrc
```

- 맨 아래 부분에 다음과 같이 입력 후 저장하고 나온다.

```
# set CUDA_PATH
export CUDA_HOME=/usr/local/cuda-9.0
export PATH=$CUDA_HOME/bin:$PATH
export LD_LIBRARY_PATH=$CUDA_HOME/lib64:$LD_LIBRARY_PATH
```

# cuDNN 설치

- 파일을 다운로드 받은 후 압축을 푼다.

```
$ tar xvzf cudnn-9.1-linux-x64-v7
```

- 라이브러리 파일들을 /usr/local/cuda 에 복사한다.

```
$ sudo chmod a+r cuda/include/cudnn.h
$ sudo chmod a+r cuda/lib64/libcudnn*
$ sudo cp cuda/include/cudnn.h /usr/local/cuda-9.1/include/
$ sudo cp cuda/lib64/libcudnn* /usr/local/cuda-9.1/lib64/
```

# 대망의 TensorFlow 설치

```
$ pip install tensorflow-gpu
```

또는

```
$ pip install tensorflow
```

# TensorFlow

# What is TensorFlow?

- 구글 브레인팀이 만든 딥러닝(머신러닝) framework
- Open source software library
- Data flow 그래프를 이용하여 numerical 계산을 수행
- 다른 분야에도 폭넓게 적용할 수 있을 만큼 굉장히 일반적임

# Why TensorFlow?

- Python!! (C++, Java, R, Go)
- Multiple platform: one or more CPUs or GPUs in a desktop, server, or mobile device with a single API
- Flexibility: from Raspberry Pi, Android, Windows, iOS, Linux to server farms
- Visualization (TensorBoard)
- Auto-differentiation autodiff
- Large community and awesome projects already using TensorFlow
- Software that includes TensorFlow, TensorBoard, and TensorServing

# Compare Others

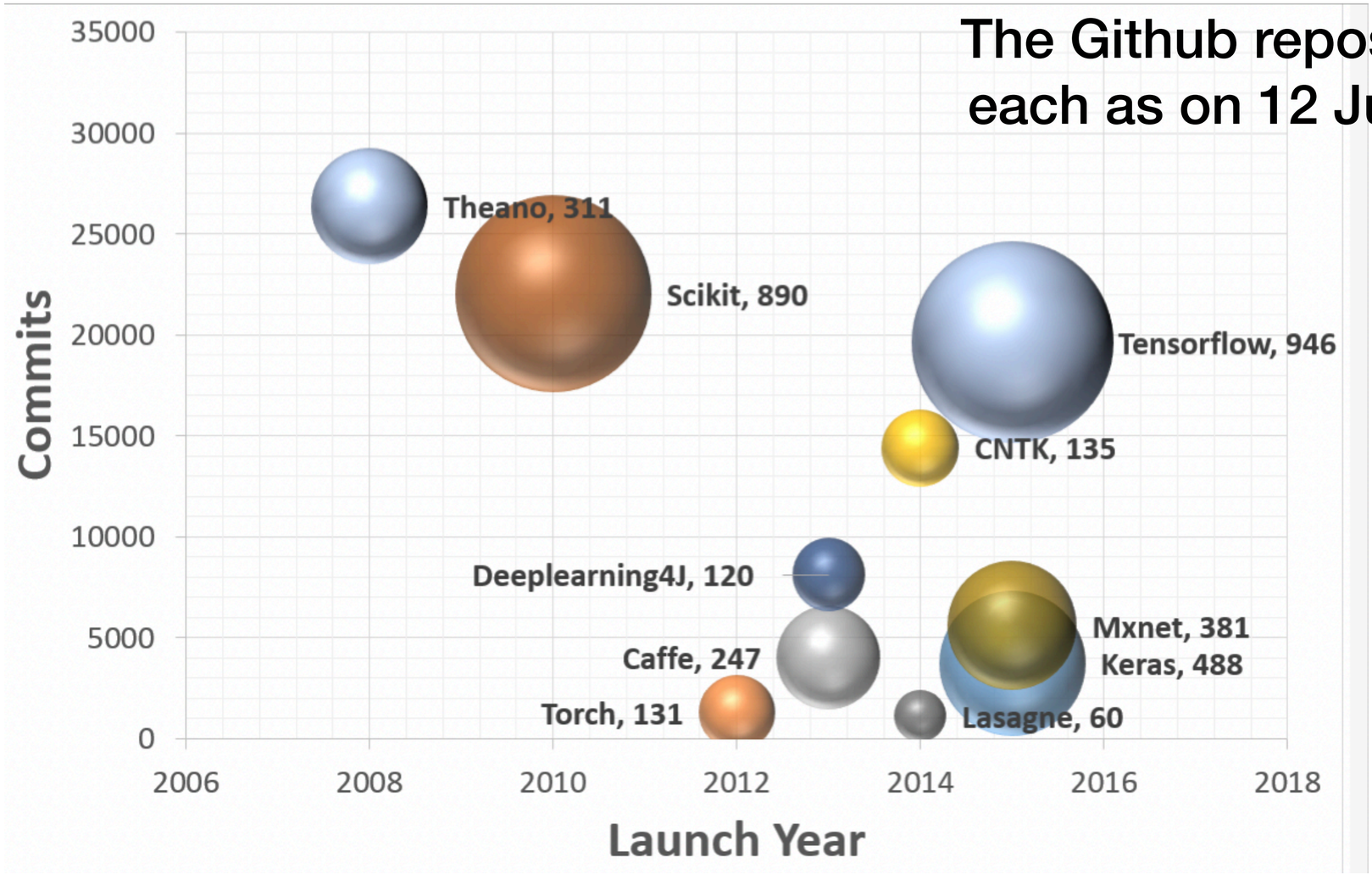
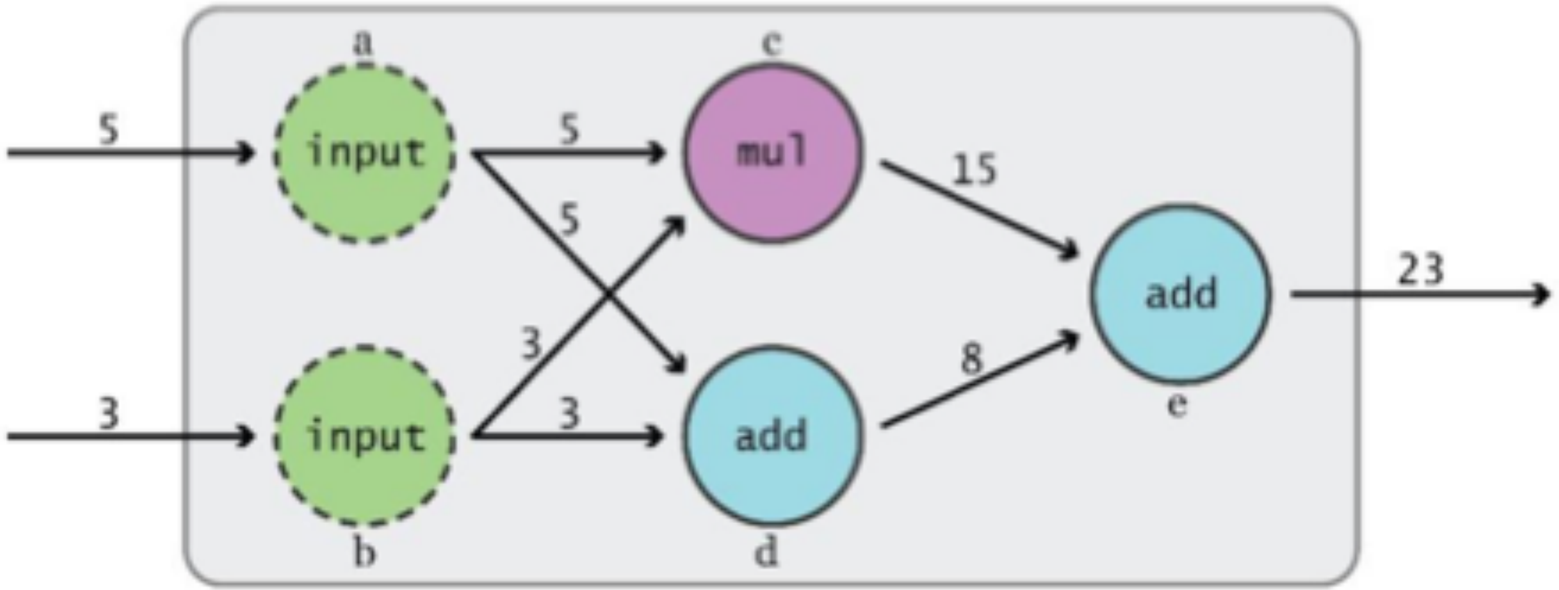



Figure credit: TensorFlow 1.x Deep Learning Cookbook

# Graphs and Sessions

# Data Flow Graphs

- Phase 1: 그래프를 구성한다 (정적 그래프 방식)
- Phase 2: session을 이용하여 그래프의 각 연산을 수행한다



# What is a Tensor?

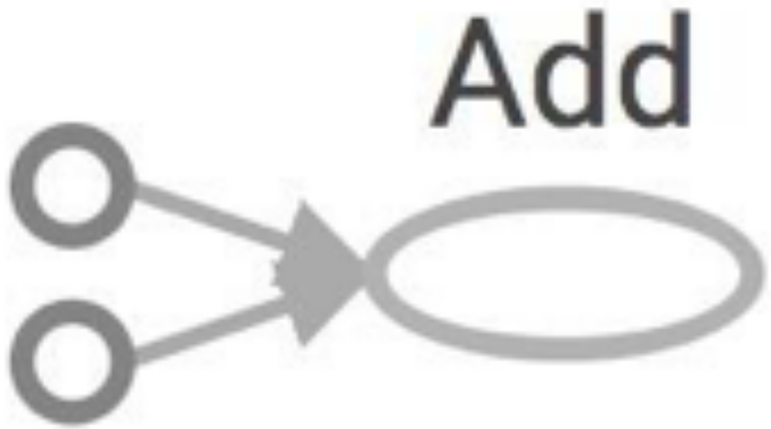
- An n-dimensional array
  - rank 0: scalar
  - rank 1: vector
  - rank 2: matrix
  - rank 3: 3-tensor (cube 모양)
  - rank n: n-tensor

# Data Flow Graphs

```
In [2]: import tensorflow as tf

In [3]: a = tf.add(3, 5)
```

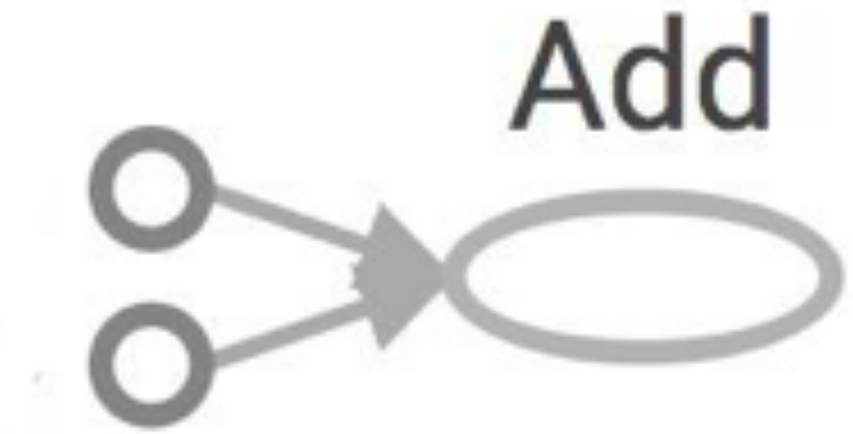
# Data Flow Graphs

```
In [2]: import tensorflow as tf

In [3]: a = tf.add(3, 5)

In [4]: print(a)

Tensor("Add:0", shape=(), dtype=int32)
```



# Data Flow Graphs

```
In [2]: import tensorflow as tf

In [3]: a = tf.add(3, 5)

In [4]: print(a)

Tensor("Add:0", shape=(), dtype=int32)
```




파이프를 연결하여 길을 만드는 과정

# Session

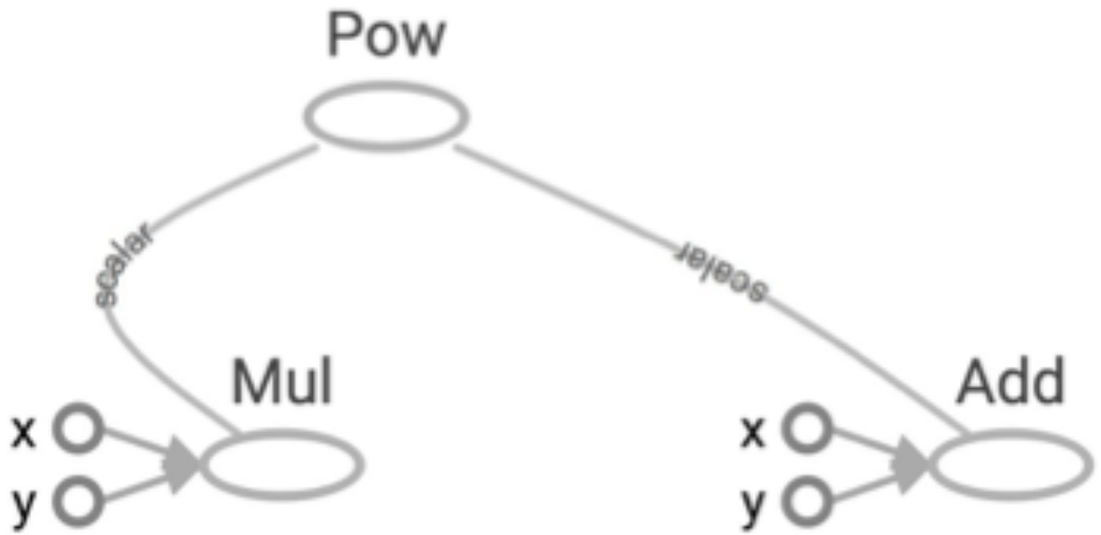
```
In [2]: import tensorflow as tf

In [3]: a = tf.add(3, 5)

In [4]: print(a)

        Tensor("Add:0", shape=(), dtype=int32)

In [5]: sess = tf.Session()
        print(sess.run(a))

        8
```


파이프에 물(데이터)을 흘려보내서 실제 계산을 수행

# More Graphs

In [7]:
```
x = 2
y = 3
w = tf.add(x, y)
z = tf.multiply(x, y)
p = tf.pow(z, w)
with tf.Session() as sess:
  print(sess.run(p))
```

```
7776
```



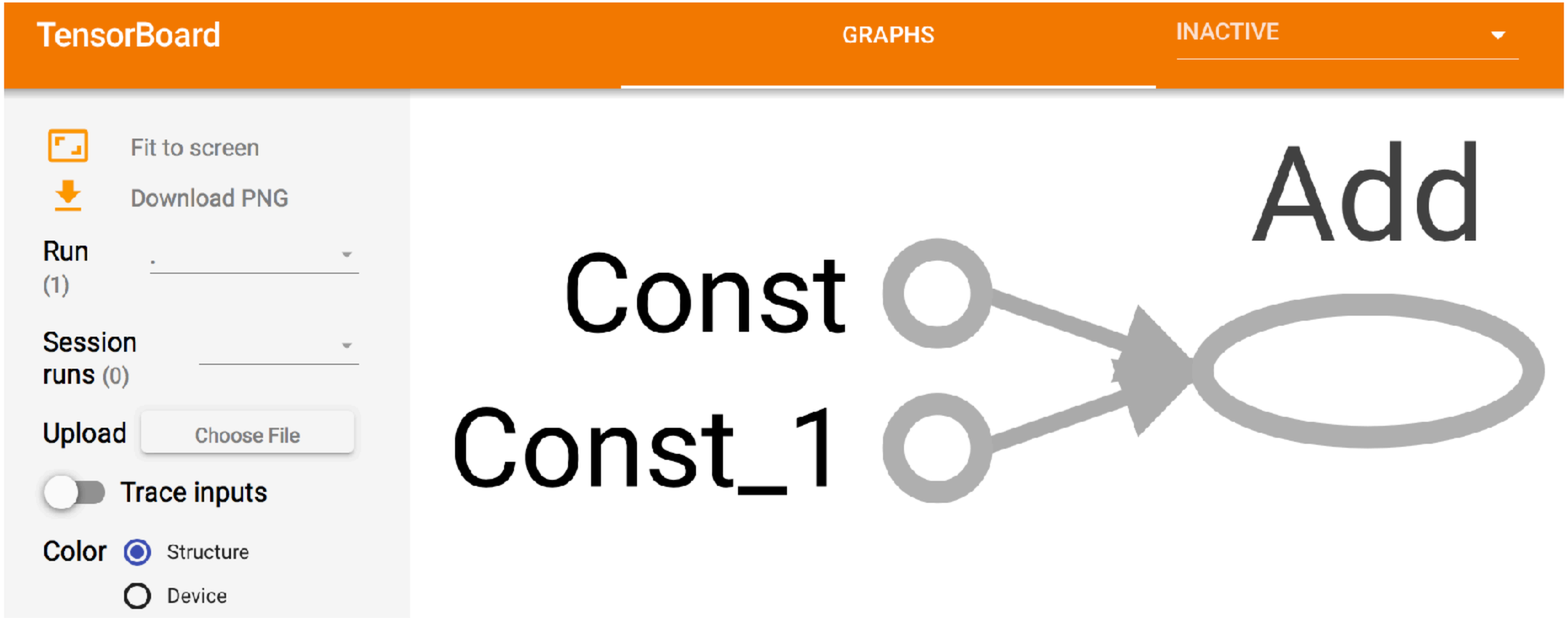
# Tensorboard

In [3]:
```
a = tf.constant(2)
b = tf.constant(3)
x = tf.add(a, b)
with tf.Session() as sess:
  # add this line to use TensorBoard.
  writer = tf.summary.FileWriter("./graphs", sess.graph)
  print(sess.run(x))
writer.close() # close the writer when you're done using it
```

```
5
```

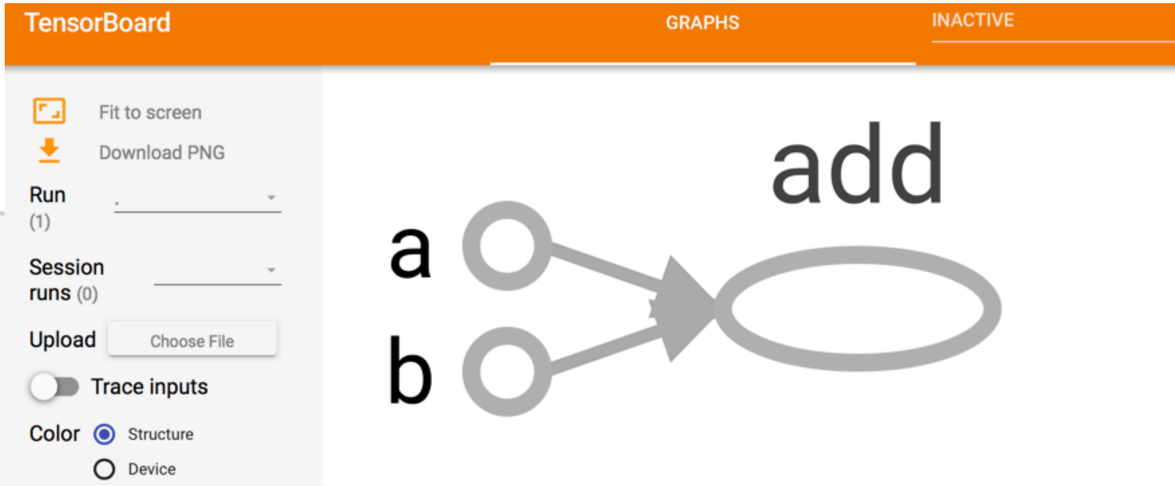
# Tensorboard

```
$ tensorboard –logdir graphs
브라우저 localhost:6006
```



# Explicitly Name

```
In [3]: a = tf.constant(2, name='a')
        b = tf.constant(3, name='b')
        x = tf.add(a, b, name='add')
        with tf.Session() as sess:
          # add this line to use TensorBoard.
          writer = tf.summary.FileWriter("./graphs", sess.graph)
          print(sess.run(x))
        writer.close() # close the writer when you're done using it

        5
```



# TensorFlow Data Dimensions

code03_TF_dimension.ipynb

# Three Types of Tensors

- `tf.constant`
- `tf.Variable`
- `tf.placeholder`

# tf.Variable

- 신경망에서 가중치와 같은 학습 가능한 parameter를 정의 할 때나 코드가 실행될 때 값이 변경 될 사항이 있을 때 유용함

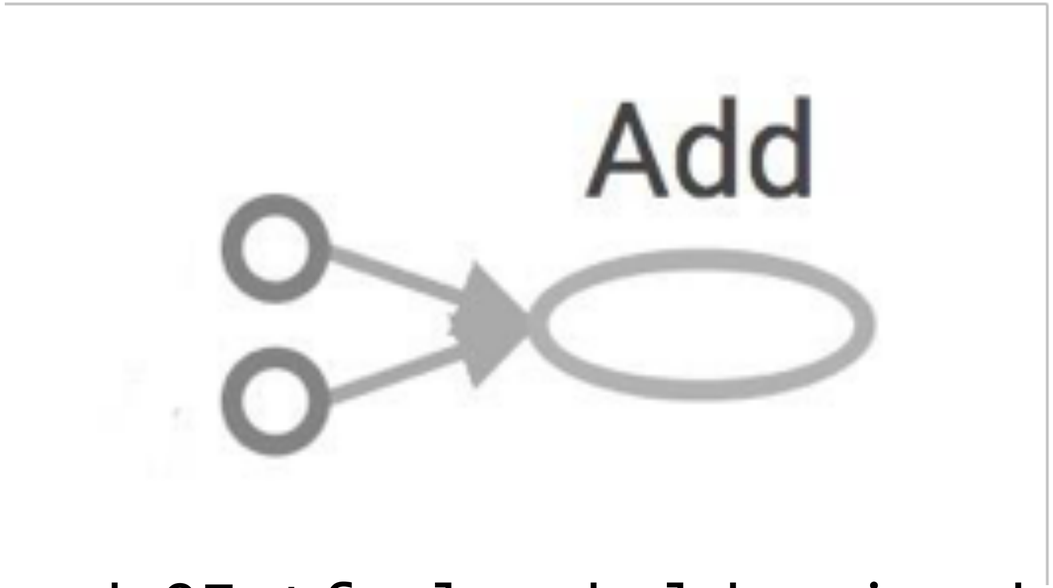
- cf) `tf.constant`: session을 통해 실행될 때 값이 변하지 않음 (immutable)

`code04_tf.Variable.ipynb`

# tf.placeholder

- tf.Session() 을 실행 할 때 외부에서 값을 넣어줌

- 학습데이터 또는 추론(inference) 할 때의 개별 데이터처럼 그래프 외부에서 값을 넣어 주는 형태로 만들 필요가 있을 때 유용함




code05_tf.placeholder.ipynb

# tf.train.Saver

- Variable을 저장 또는 불러오기 위해 사용한다.
- 중요한건 tensorflow 변수 이름이다.

code06_tf.train.Saver.ipynb