작업 환경 설정

딥러닝 학습을 위한 작업 및 개발 환경의 설정

권장 시스템

- 적정 사양의 데스크탑 PC
- NVIDIA Graphics Card
- 우분투 리눅스 14.04 (Ubuntu Linux 14.04)

우분투 리눅스 설치

- 권장 OS: `Ubuntu 14.04`
- 프로그램 설치 방법: `apt-get update && apt-get install`
- NVIDIA 드라이버: `nvidia-384` (2017.8.21 현재)
- 기본 설치 패키지: `build-essential`

우분투 리눅스 - 기본

- root user vs. normal user
  - 우분투 리눅스에서는 `root` 사용자로 로그인 하는 기능을 권장하지 않음
  - `sudo` 명령으로 `root` 사용자와 동일한 권한으로 실행
  - `root` 사용자 권한으로 실행시 사소한 오타로 시스템 삭제가 가능하므로 신중하게

우분투 리눅스 - 기본 - 패키지관리

- 패키지 목록 업데이트 (인터넷 서버에서 가져옴)

```
sudo apt-get update
```

- 패키지 이름으로 검색

```
apt-cache search ^nvidia-
```

apt-get update 실행의 예시

```
user01@94451723dfcc:~$ sudo apt-get update
[sudo] password for user01:
Ign http://archive.ubuntu.com trusty InRelease
Get:1 http://archive.ubuntu.com trusty-updates InRelease [65.9 kB]
Get:2 http://archive.ubuntu.com trusty-security InRelease [65.9 kB]
Hit http://archive.ubuntu.com trusty Release.gpg
Get:3 http://archive.ubuntu.com trusty-updates/main Sources [498 kB]
Get:4 http://archive.ubuntu.com trusty-updates/restricted Sources [6470 B]
Get:5 http://archive.ubuntu.com trusty-updates/universe Sources [237 kB]
Get:6 http://archive.ubuntu.com trusty-updates/main amd64 Packages [1261 kB]
Get:7 http://archive.ubuntu.com trusty-updates/restricted amd64 Packages [21.2 kB]
Get:8 http://archive.ubuntu.com trusty-updates/universe amd64 Packages [543 kB]
Get:9 http://archive.ubuntu.com trusty-security/main Sources [178 kB]
Get:10 http://archive.ubuntu.com trusty-security/restricted Sources [5068 B]
Get:11 http://archive.ubuntu.com trusty-security/universe Sources [73.5 kB]
Get:12 http://archive.ubuntu.com trusty-security/main amd64 Packages [812 kB]
Get:13 http://archive.ubuntu.com trusty-security/restricted amd64 Packages [17.9 kB]
Get:14 http://archive.ubuntu.com trusty-security/universe amd64 Packages [230 kB]
Hit http://archive.ubuntu.com trusty Release
Hit http://archive.ubuntu.com trusty/main Sources
Hit http://archive.ubuntu.com trusty/restricted Sources
Hit http://archive.ubuntu.com trusty/universe Sources
Hit http://archive.ubuntu.com trusty/main amd64 Packages
Hit http://archive.ubuntu.com trusty/restricted amd64 Packages
Hit http://archive.ubuntu.com trusty/universe amd64 Packages
Fetched 4015 kB in 9s (436 kB/s)
Reading package lists... Done
user01@94451723dfcc:~$
```

apt-cache search 의 예시

```
user01@94451723dfcc:~$ apt-cache search ^nvidia-3
nvidia-304-dev - NVIDIA binary Xorg driver development files
nvidia-310 - Transitional package for nvidia-310
nvidia-310-dev - Transitional package for nvidia-310-dev
nvidia-310-updates - Transitional package for nvidia-310-updates
nvidia-310-updates-dev - Transitional package for nvidia-310-updates-dev
nvidia-313-updates - Transitional package for nvidia-313-updates
nvidia-313-updates-dev - Transitional package for nvidia-313-updates-dev
nvidia-319 - Transitional package for nvidia-319
nvidia-319-dev - Transitional package for nvidia-319-dev
nvidia-319-updates - Transitional package for nvidia-319-updates
nvidia-319-updates-dev - Transitional package for nvidia-319-updates-dev
nvidia-304 - NVIDIA legacy binary driver - version 304.135
nvidia-304-updates - Transitional package for nvidia-304
nvidia-304-updates-dev - Transitional package for nvidia-304-dev
nvidia-331 - Transitional package for nvidia-331
nvidia-331-dev - Transitional package for nvidia-340-dev
nvidia-331-updates - Transitional package for nvidia-340
nvidia-331-updates-dev - Transitional package for nvidia-340-dev
nvidia-331-updates-uvm - Transitional package for nvidia-340
nvidia-331-uvm - Transitional package for nvidia-340
nvidia-340 - NVIDIA binary driver - version 340.102
nvidia-340-dev - NVIDIA binary Xorg driver development files
nvidia-340-updates - Transitional package for nvidia-340
nvidia-340-updates-dev - Transitional package for nvidia-340-dev
nvidia-340-updates-uvm - Transitional package for nvidia-340-updates
nvidia-340-uvm - Transitional package for nvidia-340
nvidia-346 - Transitional package for nvidia-346
nvidia-346-dev - Transitional package for nvidia-352-dev
nvidia-346-updates - Transitional package for nvidia-346-updates
nvidia-346-updates-dev - Transitional package for nvidia-352-updates-dev
nvidia-346-updates-uvm - Transitional package for nvidia-346-updates
nvidia-346-uvm - Transitional package for nvidia-346
nvidia-352 - Transitional package for nvidia-367
nvidia-352-dev - Transitional package for nvidia-367-dev
nvidia-352-updates - Transitional package for nvidia-367
nvidia-352-updates-dev - Transitional package for nvidia-367-dev
nvidia-367 - Transitional package for nvidia-375
nvidia-367-dev - Transitional package for nvidia-375-dev
nvidia-375 - NVIDIA binary driver - version 375.66
nvidia-375-dev - NVIDIA binary Xorg driver development files
user01@94451723dfcc:~$
```

우분투 리눅스 - 기본 - 패키지관리

- 패키지 설치

```
sudo apt-get install tmux
```

- 패키지 삭제

```
sudo apt-get remove tmux
```

우분투 리눅스 - 기본 - 패키지관리

- 가끔 잘 안될 때가 있음
- 먼저 `sudo apt-get update` 를 한 다음 재시도
- 인터넷 사정으로, 우분투 배포 서버 사정으로 안될 때도 있음. (몇 시간 뒤 재시도)

우분투 리눅스 - NVIDIA 드라이버 설치

- NVIDIA 카드를 위한 기본 비디오 드라이버는 nouveau 드라이버 (open-source)
- 딥러닝을 위해 CUDA 런타임을 사용하기 위해서는 NVIDIA proprietary 버젼 (non open-source) 드라이버 설치 필요

```
sudo apt-get update
sudo apt-get install nvidia-384
```

- `nvidia-384` 드라이버를 찾을 수 없다는 메시지가 나오는 경우 아래와 같이 **PPA** (Personal Package Archives) repository 추가 후 재시도

```
sudo apt-get install software-properties-common
sudo add-apt-repository ppa:graphics-drivers/ppa
sudo apt-get update
```

우분투 리눅스 - NVIDIA 드라이버 설치

- `nvidia-384` 드라이버 설치 중, 아래와 같은 에러가 발생하는 경우

```
  update-alternatives: error: error creating symbolic link `/usr/lib/
nvidia/alternate-install-present.dpkg-tmp': No such file or directory
```

  - 아래와 같은 임시 조치 후 재시도

    sudo mkdir -p /usr/lib/nvidia

우분투 리눅스 - CUDA Toolkit 파일 설치

- `cuda-repo-ubuntu1404-8-0-local-ga2_8.0.61-1_amd64.deb` 파일
- `cuda-repo-ubuntu1404-8-0-local-cublas-performance-update_8.0.61-1_amd64` 파일
  - NVIDIA 에서 제공하는 CUDA SDK https://developer.nvidia.com/cuda-downloads (https://developer.nvidia.com/cuda-downloads)

- 저장소 파일 설치

```
  # cuda-repo-ubuntu1404-8-0-local-ga2_8.0.61-1_amd64.deb
  # cuda-repo-ubuntu1404-8-0-local-cublas-performance-update_8.0.61-1
_amd64.deb
  sudo dpkg -i cuda-*.deb

  sudo apt-get update
  sudo apt-get install cuda
```

우분투 리눅스 - CUDNN5, CUDNN6 설치

- **cudnn5** - tensorflow-1.1 이전 버젼 설치에 필요
- **cudnn6** - tensorflow-1.2 이후 버젼 설치에 필요
- .deb 파일을 이용한 설치

```
# libcudnn5_5.1.5-1+cuda8.0_amd64.deb
# libcudnn5-dev_5.1.5-1+cuda8.0_amd64.deb
# libcudnn6_6.0.21-1+cuda8.0_amd64.deb
# libcudnn6-dev_6.0.21-1+cuda8.0_amd64.deb
sudo dpkg -i libcudnn*.deb
```

파이썬 개발 환경 설치

- build-essential, git
- Anaconda
- Jupyter notebook
- Docker

파이썬 개발 환경 설치 - build-essential, git

- `build-essential`: c/c++ 컴파일러 및 각종 빌드 도구들 포함
- 그외에도...
  - wget : http 방식 파일 다운로드 툴
  - curl : 또 다른 http 방식 파일 다운로드 툴
  - rsync : PC 간 파일 복사 (동기화)
- `git`: 소스코드 관리 도구 (형상관리)
- 설치:

```
sudo apt-get install build-essential wget curl rsync git
```

파이썬 개발 환경 설치 - Anaconda

- 설치:

  https://docs.continuum.io/anaconda/install/linux (`https://docs.continuum.io/anaconda/install/linux`)

  http://192.168.0.14/airi400-shared-materials/week1-day1-install-files/raw/master/Anaconda2-4.4.0-Linux-x86_64 (http://192.168.0.14/airi400-shared-materials/week1-day1-install-files/raw/master/Anaconda2-4.4.0-Linux-x86_64)

- 이걸 쓰면 뭐가 좋은가:
  - 필요한 솔루션 마다 파이썬 실행환경을 별도로 관리
  - 여러개의 파이썬 버젼을 모두 설치해서 필요에 따라 선택 실행
  - 시스템에 설치된 파이썬 바이너리 및 라이브러리를 변경해서 개발 프로젝트와 관계없는 기본 기능에서 버젼 불일치로 오류가 발생하는 일이 없어짐

파이썬 개발 환경 설치 - Anaconda - 기본 사용법

- 환경 생성:

```
conda create -n tensorflow python=2.7
```

- 환경 적용:

```
. activate tensorflow
(tensorflow) pip install --upgrade tensorflow_gpu
```

- 환경 생성 - cloning

```
conda create -n tensorflow-with-librosa --clone tensorflow
```

파이썬 개발 환경 설치 - Anaconda - 기본 사용법

- 환경 목록 확인:

```
conda env list
```

- 환경 삭제:

```
conda env remove -n tensorflow-with-librosa
```

파이썬 개발 환경 설치 - Jupyter notebook

- 설치:

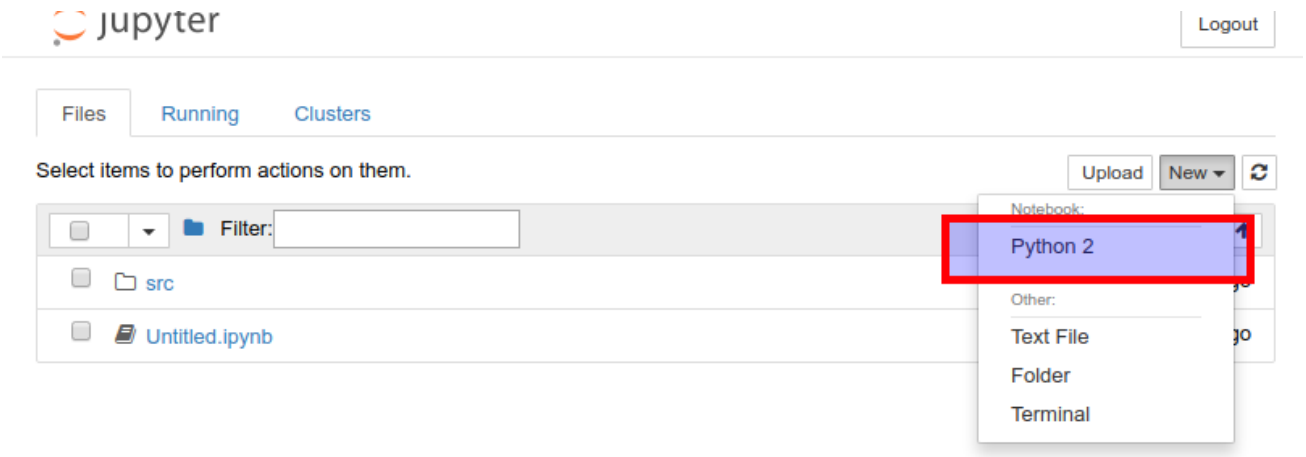
```
pip install jupyter
```

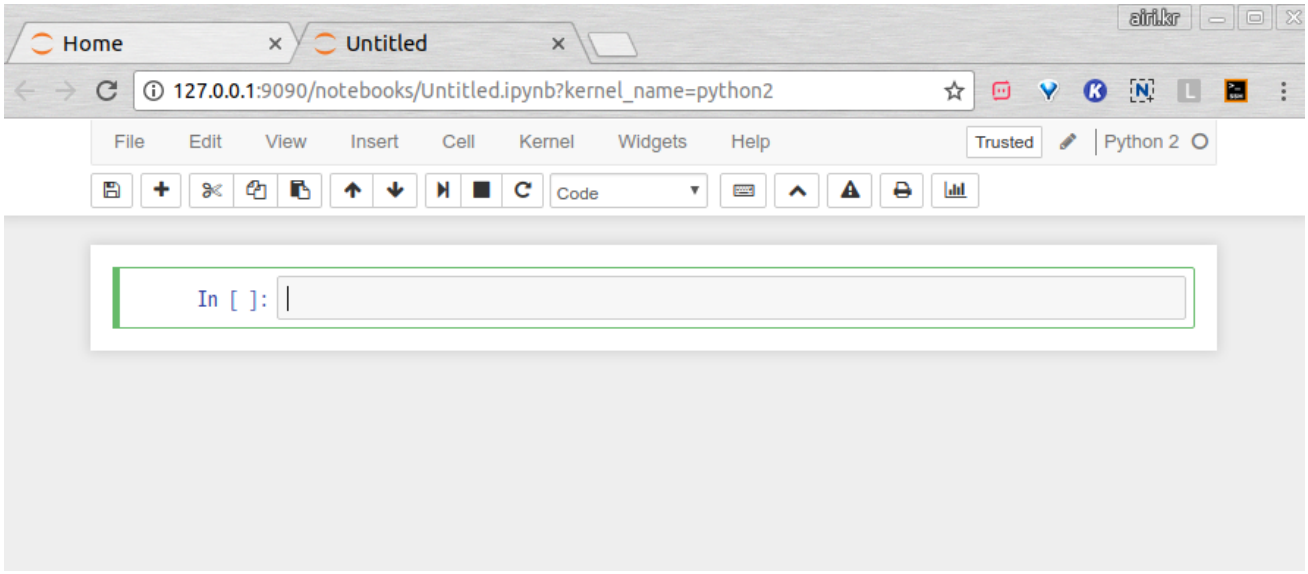
- 실행:

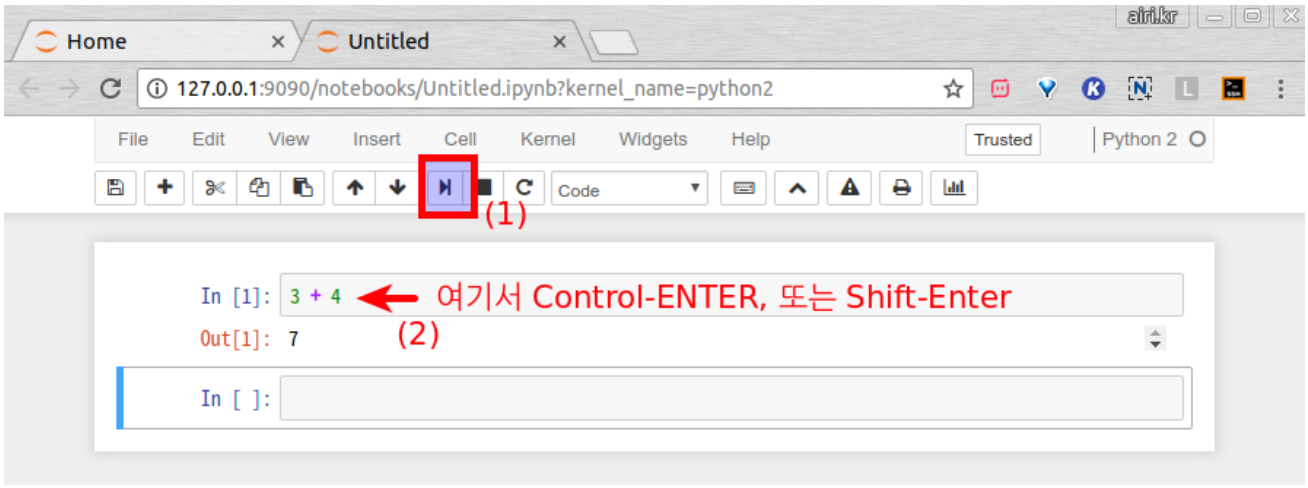
```
jupyter notebook
```

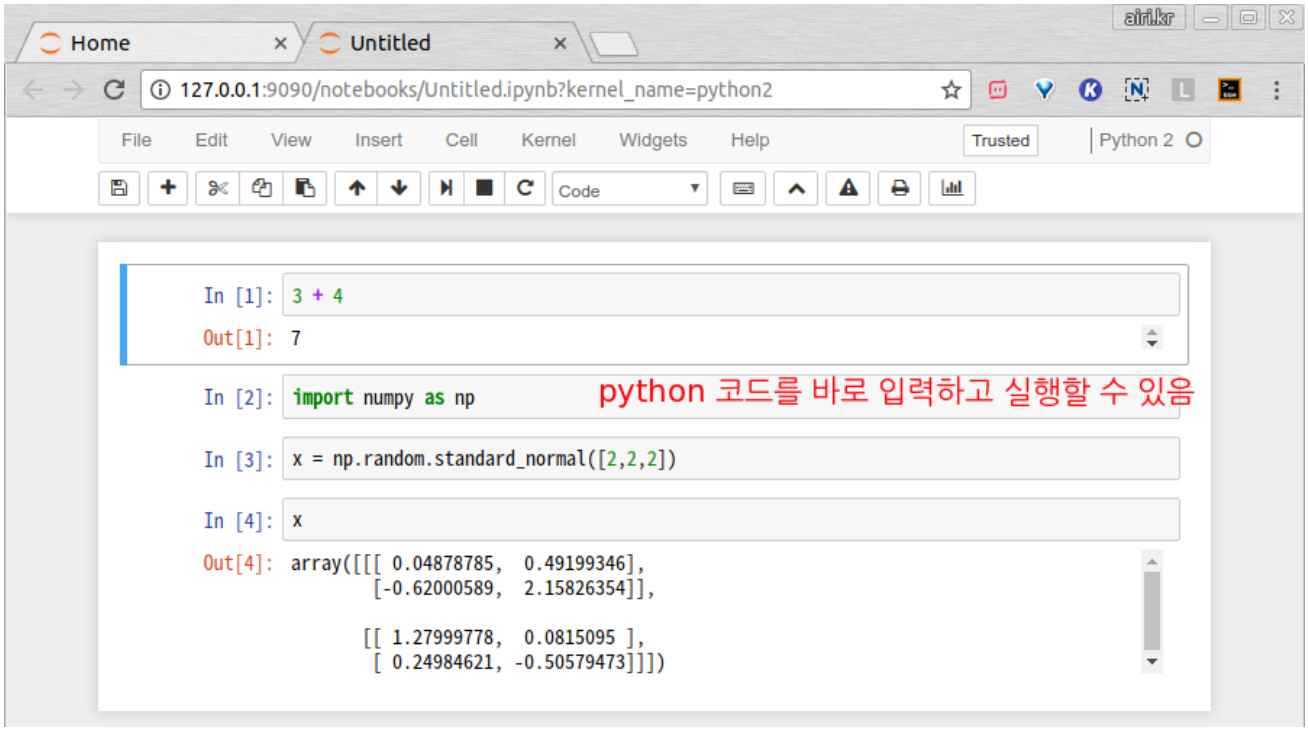
- 사용:
  - 인터넷 브라우저 (구글 크롬 등) 열어서 http://127.0.0.1:8888/ (http://127.0.0.1:8888/) 접속

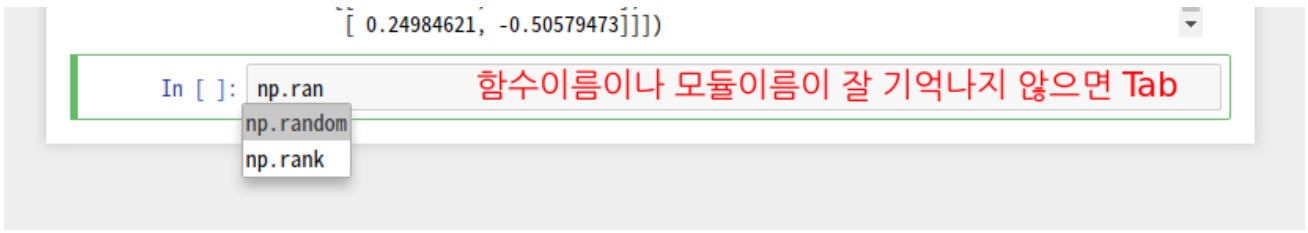
파이썬 개발 환경 설치 - Jupyter notebook - 기본 사용법

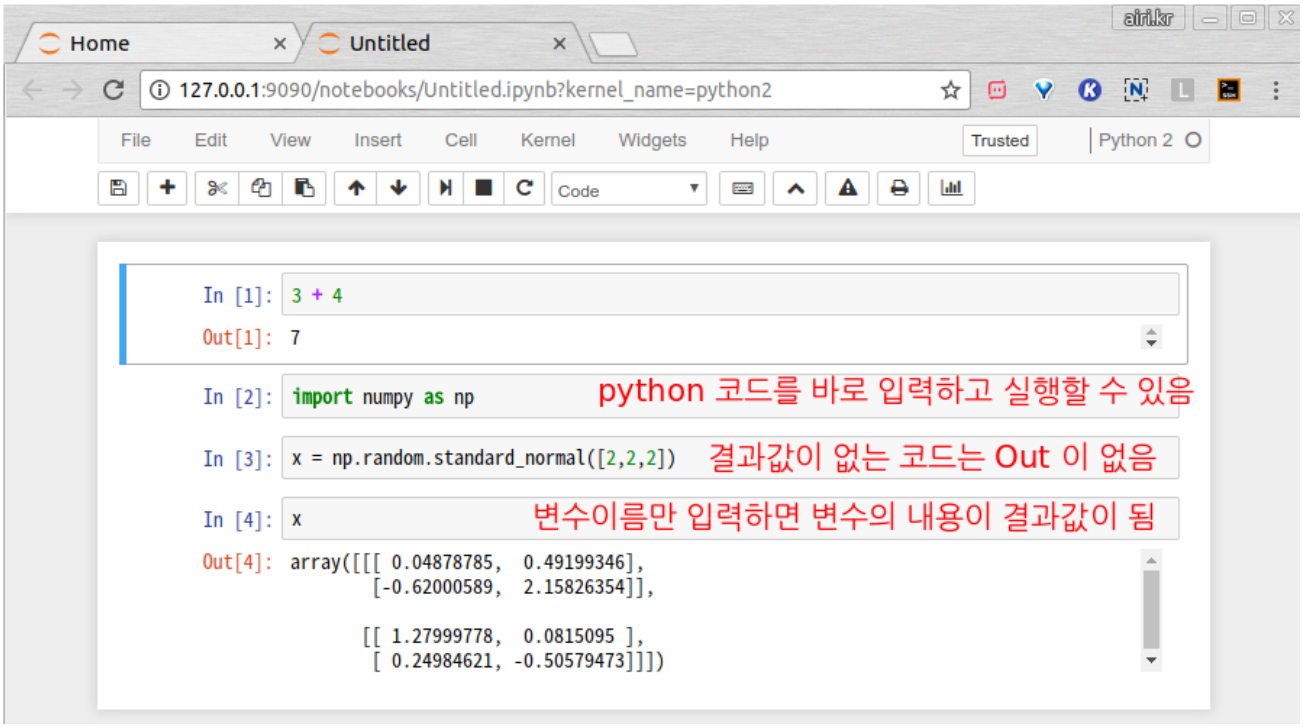


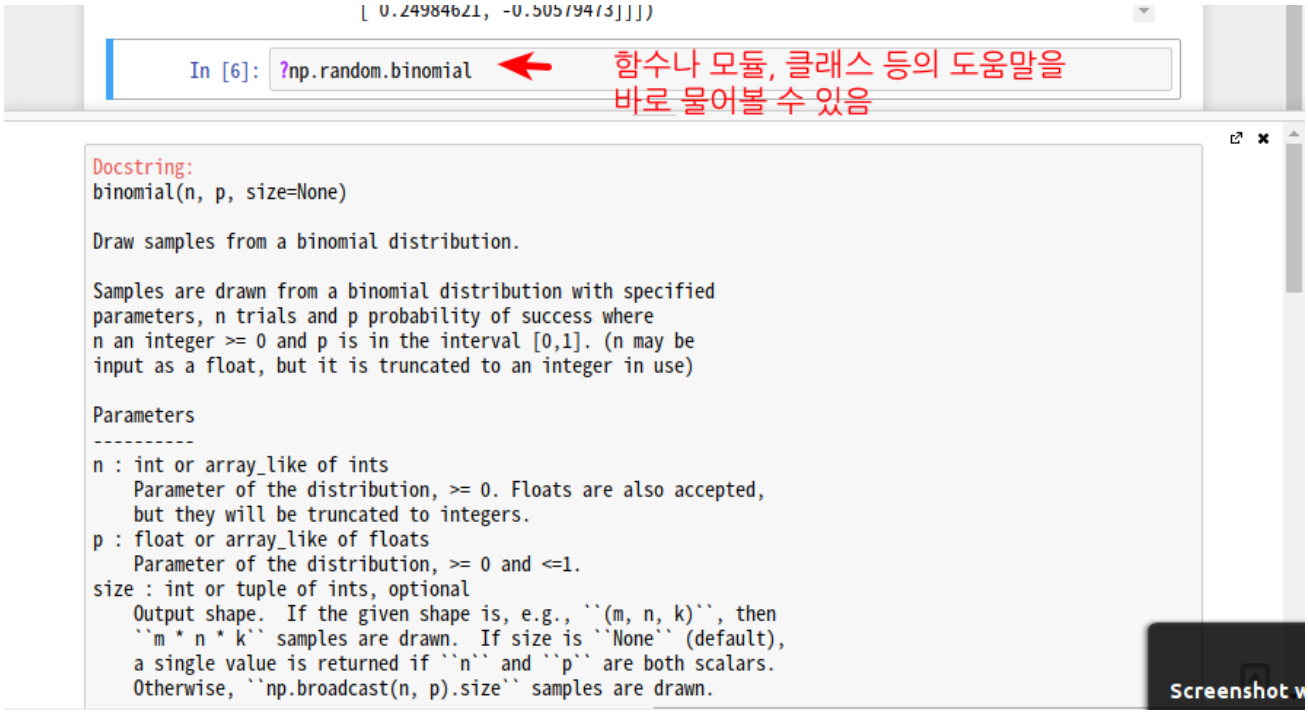

Home
Untitled
127.0.0.1:9090/notebooks/Untitled.ipynb?kernel_name=python2
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 2
Code
In [ ]:

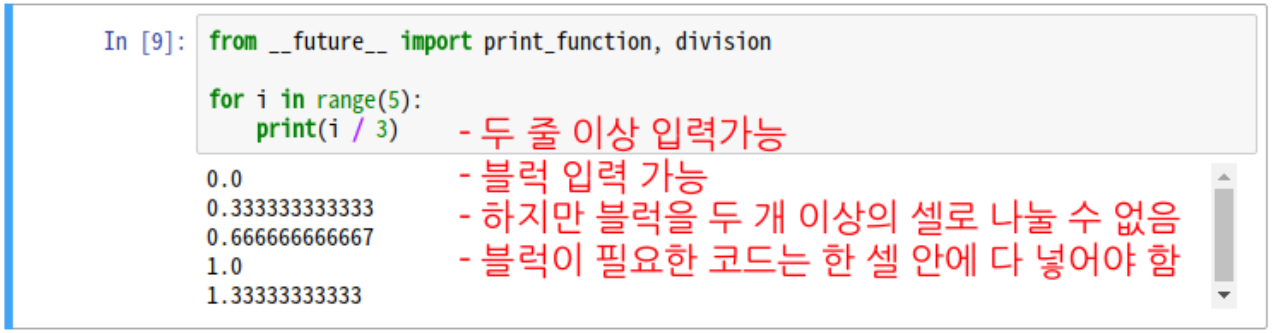

Home
Untitled
127.0.0.1:9090/notebooks/Untitled.ipynb?kernel_name=python2
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 2
Code
(1)
In [1]: 3 + 4
여기서 Control-ENTER, 또는 Shift-Enter
(2)
Out[1]: 7
In [ ]:

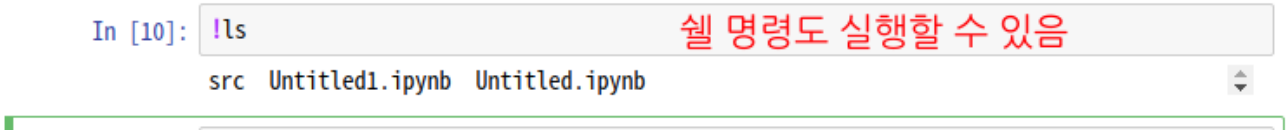

Home
Untitled
127.0.0.1:9090/notebooks/Untitled.ipynb?kernel_name=python2
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 2
Code
In [1]: 3 + 4
Out[1]: 7
In [2]: import numpy as np
python 코드를 바로 입력하고 실행할 수 있음
In [3]: x = np.random.standard_normal([2,2,2])
In [4]: x
Out[4]: array([[[0.04878785, 0.49199346],
[-0.62000589, 2.15826354]],
[[1.27999778, 0.0815095],
[0.24984621, -0.50579473]]])


[0.24984621, -0.50579473]]])
In [ ]: np.ran
함수이름이나 모듈이름이 잘 기억나지 않으면 Tab
np.random
np.rank


Home
Untitled
127.0.0.1:9090/notebooks/Untitled.ipynb?kernel_name=python2
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 2
Code
In [1]: 3 + 4
Out[1]: 7
In [2]: import numpy as np
python 코드를 바로 입력하고 실행할 수 있음
In [3]: x = np.random.standard_normal([2,2,2])
결과값이 없는 코드는 Out 이 없음
In [4]: x
변수이름만 입력하면 변수의 내용이 결과값이 됨
Out[4]: array([[[0.04878785, 0.49199346],
[-0.62000589, 2.15826354]],
[[1.27999778, 0.0815095],
[0.24984621, -0.50579473]]])


In [6]: ?np.random.binomial
함수나 모듈, 클래스 등의 도움말을
바로 물어볼 수 있음
Docstring:
binomial(n, p, size=None)
Draw samples from a binomial distribution.
Samples are drawn from a binomial distribution with specified
parameters, n trials and p probability of success where
n an integer >= 0 and p is in the interval [0,1]. (n may be
input as a float, but it is truncated to an integer in use)
Parameters
n : int or array_like of ints
Parameter of the distribution, >= 0. Floats are also accepted,
but they will be truncated to integers.
p : float or array_like of floats
Parameter of the distribution, >= 0 and <=1.
size : int or tuple of ints, optional
Output shape. If the given shape is, e.g., ``(m, n, k)``, then
``m * n * k`` samples are drawn. If size is ``None`` (default),
a single value is returned if ``n`` and ``p`` are both scalars.
Otherwise, ``np.broadcast(n, p).size`` samples are drawn.


In [9]:
from __future__ import print_function, division
for i in range(5):
print(i / 3)
0.0
0.333333333333
0.666666666667
1.0
1.33333333333
- 두 줄 이상 입력가능
- 블럭 입력 가능
- 하지만 블럭을 두 개 이상의 셀로 나눌 수 없음
- 블럭이 필요한 코드는 한 셀 안에 다 넣어야 함

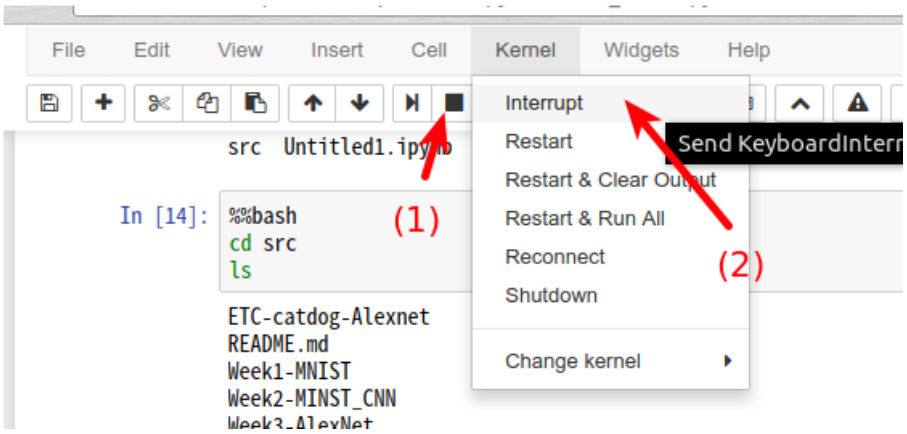

In [10]: !ls
쉘 명령도 실행할 수 있음
src Untitled1.ipynb Untitled.ipynb


In [14]:
%%bash
cd src
ls
두 줄 이상 되는 쉘 명령 시퀀스도 실행할 수 있음
ETC-catdog-Alexnet
README.md
Week1-MNIST
Week2-MINST_CNN
Week3-AlexNet
Week3-Inception-ResNet
Week5-RNN
Week6-DCGAN
Week6-WaveNet

In [*]:
```
import time

while True:
    print('Hello')
    time.sleep(1)
```

무한루프를 돌 수도 있음

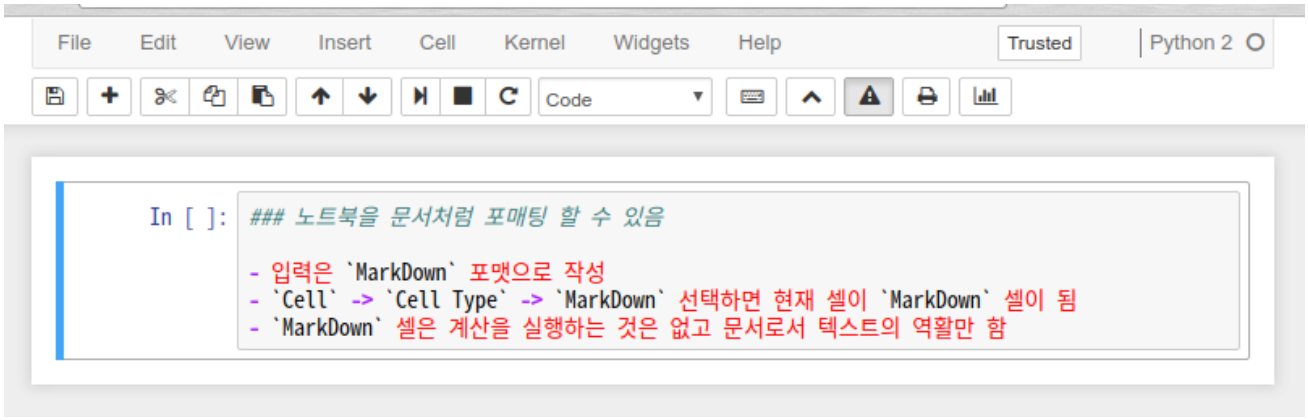
```
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
```

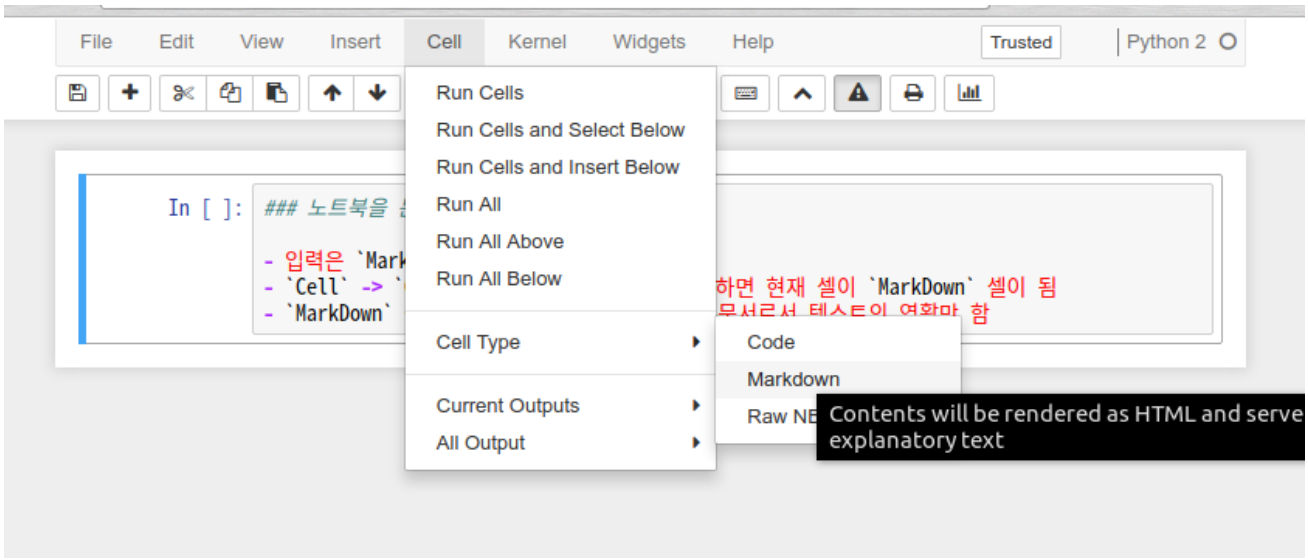


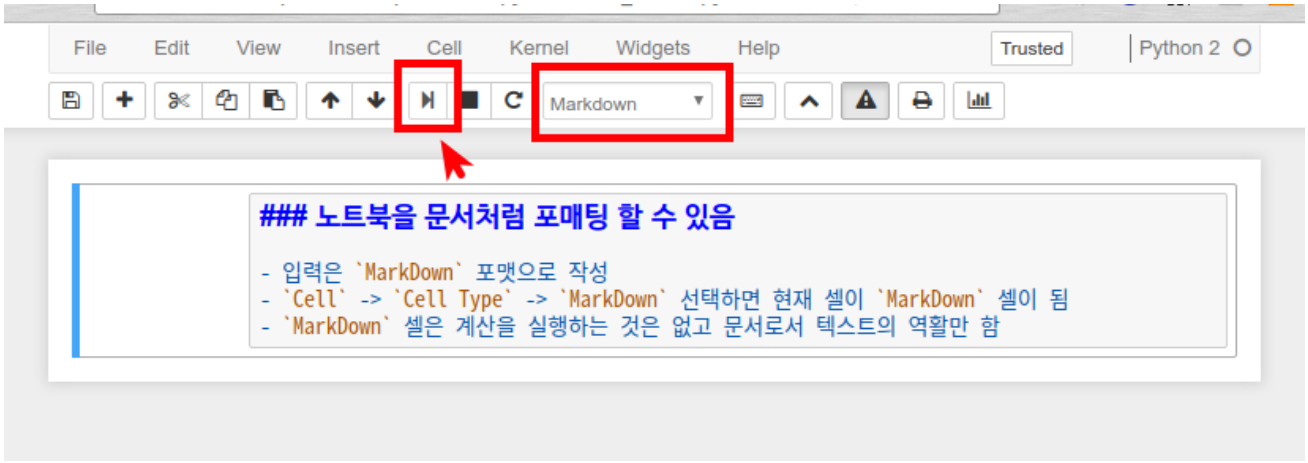
In [22]:
```
import time

while True:
    print('Hello')
    time.sleep(1)
```

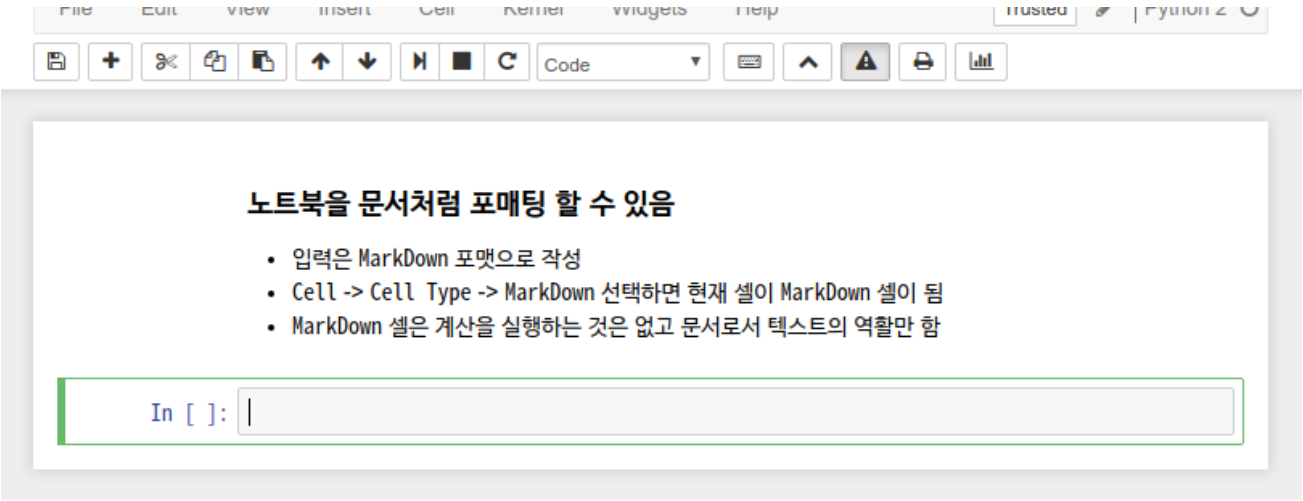
```
Hello
Hello
Hello
```

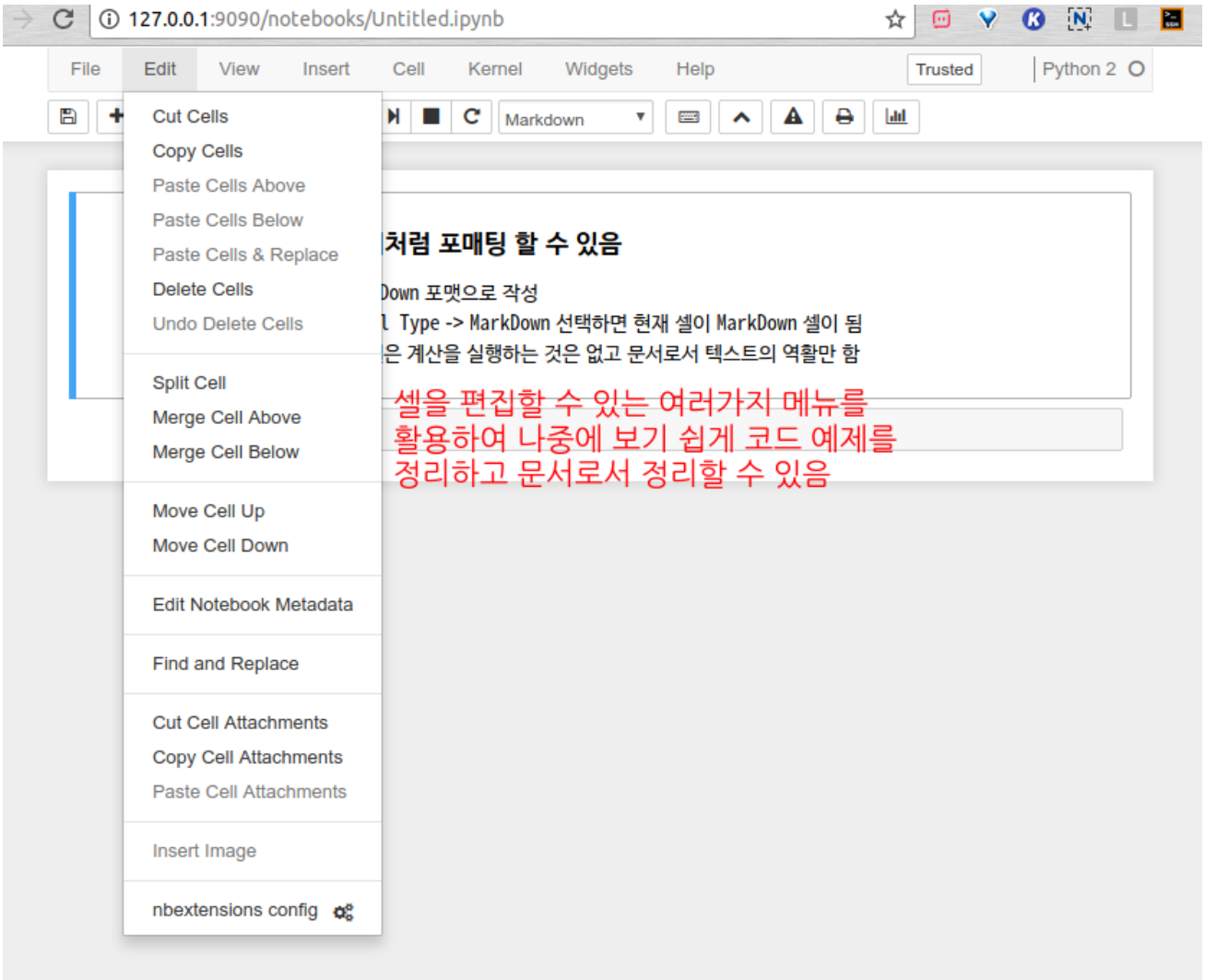
```
KeyboardInterruptTraceback (most recent call last)
<ipython-input-22-2be19f09caa0> in <module>()
      3 while True:
      4     print('Hello')
----> 5     time.sleep(1)

KeyboardInterrupt:
```

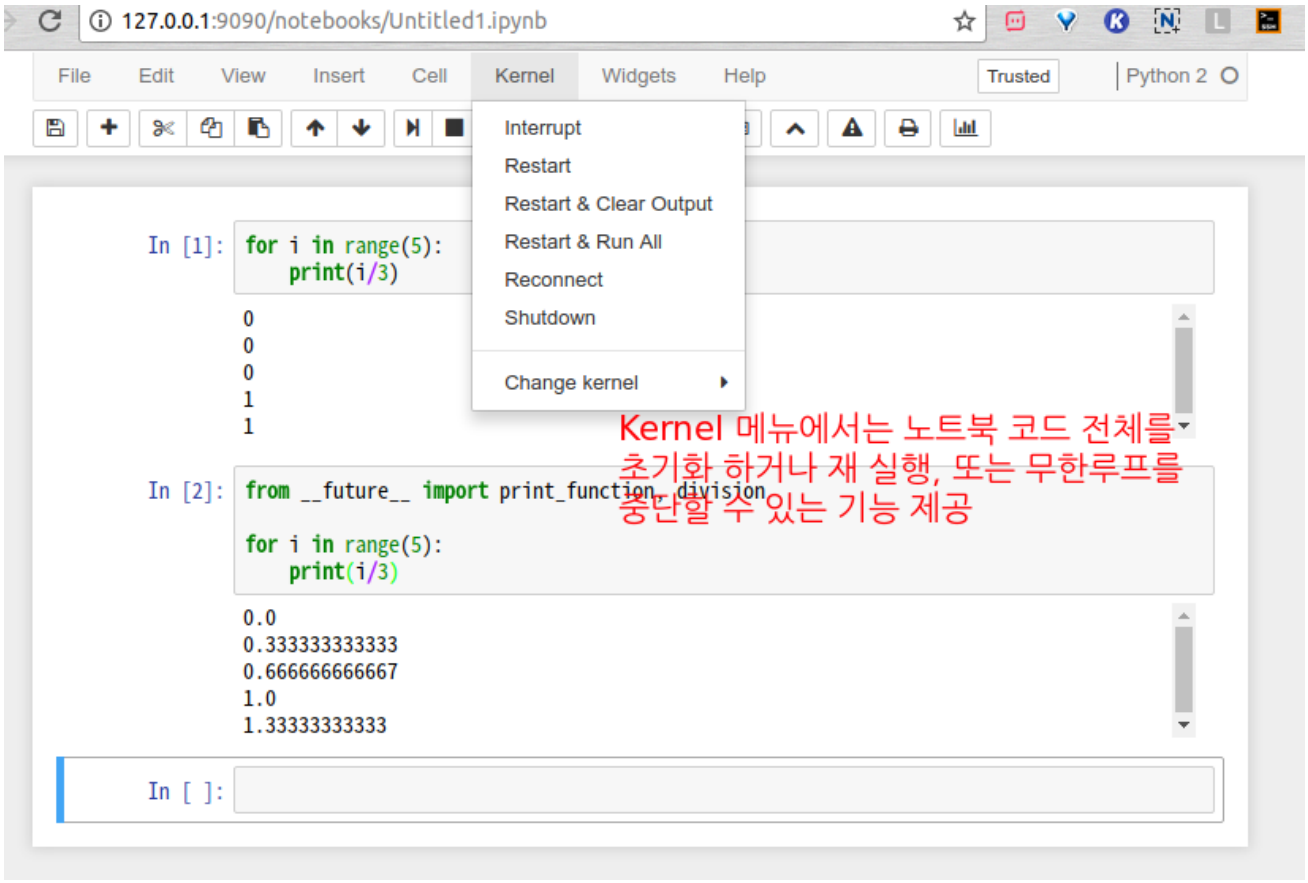

File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 2
Code
In [ ]:
노트북을 문서처럼 포매팅 할 수 있음
- 입력은 `MarkDown` 포맷으로 작성
- `Cell` -> `Cell Type` -> `MarkDown` 선택하면 현재 셀이 `MarkDown` 셀이 됨
- `MarkDown` 셀은 계산을 실행하는 것은 없고 문서로서 텍스트의 역할만 함

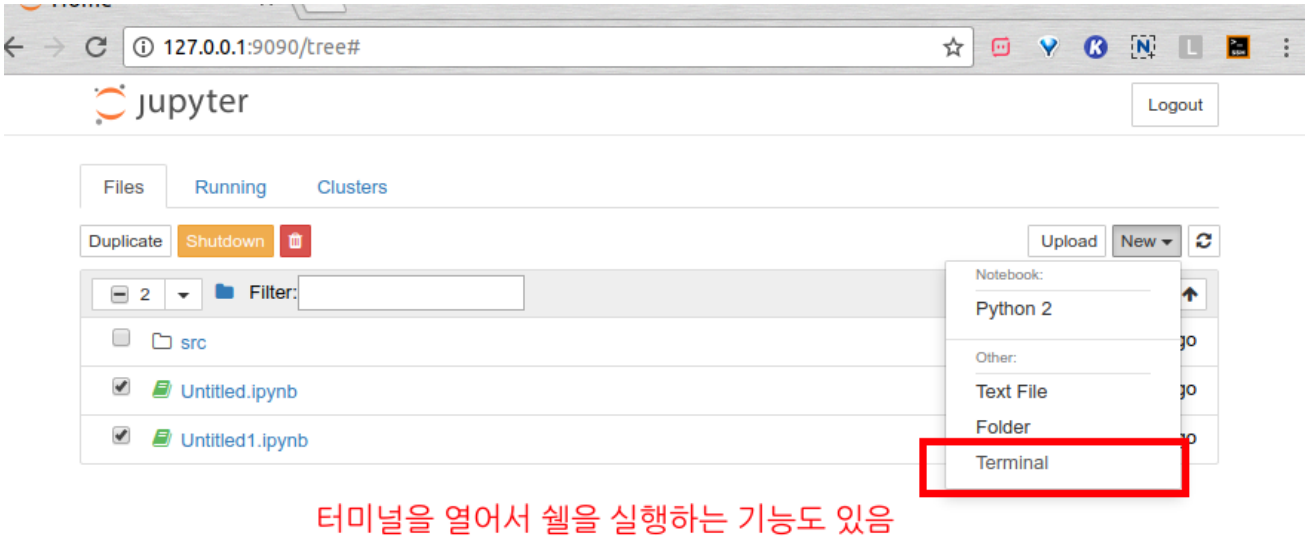

File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 2
Run Cells
Run Cells and Select Below
Run Cells and Insert Below
Run All
Run All Above
Run All Below
Cell Type
Current Outputs
All Output
Code
Markdown
Raw NB
Contents will be rendered as HTML and serve
explanatory text

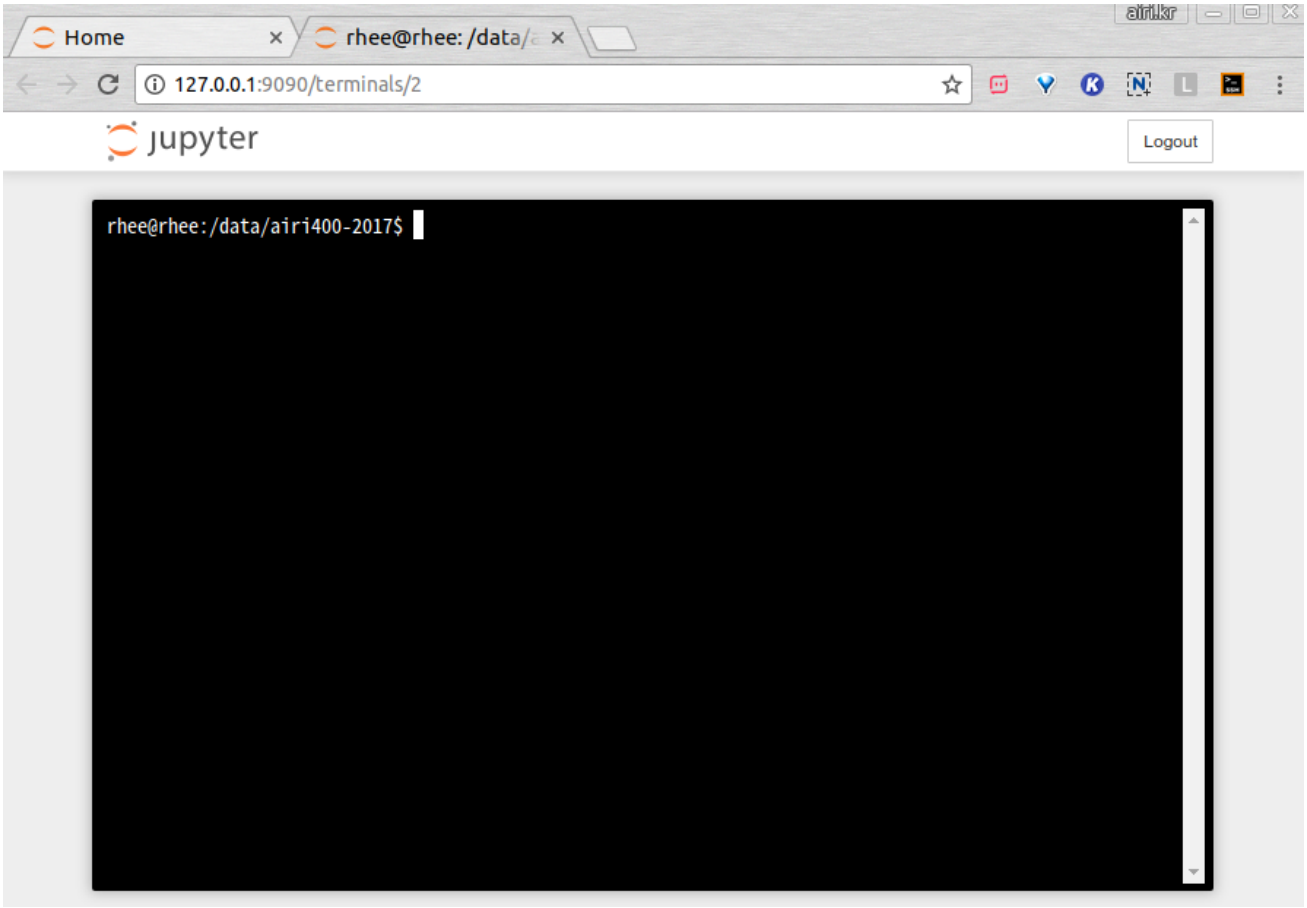

File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 2
Markdown
노트북을 문서처럼 포매팅 할 수 있음
- 입력은 `MarkDown` 포맷으로 작성
- `Cell` -> `Cell Type` -> `MarkDown` 선택하면 현재 셀이 `MarkDown` 셀이 됨
- `MarkDown` 셀은 계산을 실행하는 것은 없고 문서로서 텍스트의 역할만 함


File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 2
Code
노트북을 문서처럼 포매팅 할 수 있음
입력은 MarkDown 포맷으로 작성
Cell -> Cell Type -> MarkDown 선택하면 현재 셀이 MarkDown 셀이 됨
MarkDown 셀은 계산을 실행하는 것은 없고 문서로서 텍스트의 역활만 함
In [ ]:


127.0.0.1:9090/notebooks/Untitled.ipynb
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 2
Markdown
Cut Cells
Copy Cells
Paste Cells Above
Paste Cells Below
Paste Cells & Replace
Delete Cells
Undo Delete Cells
Split Cell
Merge Cell Above
Merge Cell Below
Move Cell Up
Move Cell Down
Edit Notebook Metadata
Find and Replace
Cut Cell Attachments
Copy Cell Attachments
Paste Cell Attachments
Insert Image
nbextensions config
처럼 포매팅 할 수 있음
Down 포맷으로 작성
l Type -> MarkDown 선택하면 현재 셀이 MarkDown 셀이 됨
은 계산을 실행하는 것은 없고 문서로서 텍스트의 역활만 함
셀을 편집할 수 있는 여러가지 메뉴를
활용하여 나중에 보기 쉽게 코드 예제를
정리하고 문서로서 정리할 수 있음


127.0.0.1:9090/notebooks/Untitled1.ipynb
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 2
Interrupt
Restart
Restart & Clear Output
Restart & Run All
Reconnect
Shutdown
Change kernel
In [1]:
for i in range(5):
print(i/3)
0
0
0
1
1
In [2]:
from __future__ import print_function, division
for i in range(5):
print(i/3)
0.0
0.333333333333
0.666666666667
1.0
1.33333333333
In [ ]:
Kernel 메뉴에서는 노트북 코드 전체를
초기화 하거나 재 실행, 또는 무한루프를
중단할 수 있는 기능 제공


127.0.0.1:9090/tree#
jupyter
Logout
Files
Running
Clusters
Duplicate
Shutdown
Upload
New
Filter:
src
Untitled.ipynb
Untitled1.ipynb
Notebook:
Python 2
Other:
Text File
Folder
Terminal
터미널을 열어서 쉘을 실행하는 기능도 있음



기타 여러가지 기능이 있음. 별도 문건 검색/참조 추천

- https://steemit.com/kr/@sanghkaang/2-jupyter (https://steemit.com/kr/@sanghkaang/2-jupyter)
- https://www.datacamp.com/community/tutorials/tutorial-jupyter-notebook#gs.8msAu8s (https://www.datacamp.com/community/tutorials/tutorial-jupyter-notebook#gs.8msAu8s)

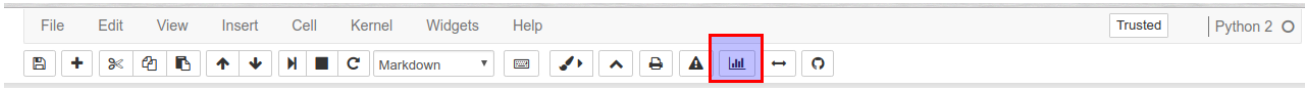
파이썬 개발 환경 설치 - Jupyter notebook - RISE

- 용도:

  - jupyter notebook 화면을 프리젠테이션 모드로 바꿔줌

- 설치:

```
pip install RISE
jupyter-nbextension install rise --py --sys-prefix
jupyter-nbextension enable rise --py --sys-prefix
```

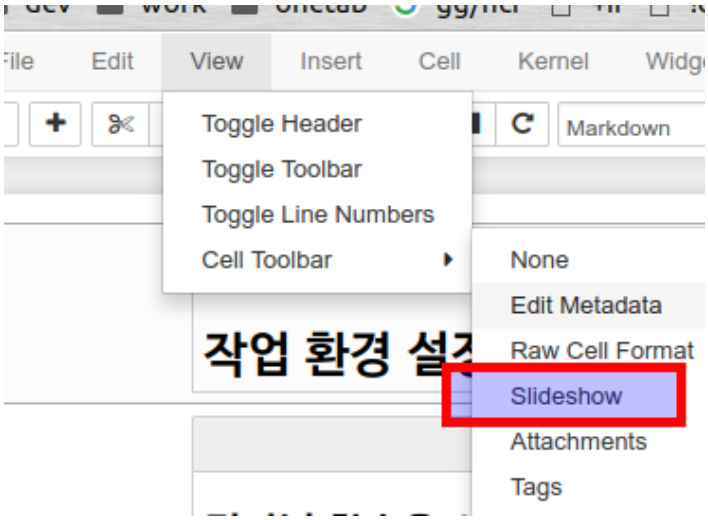


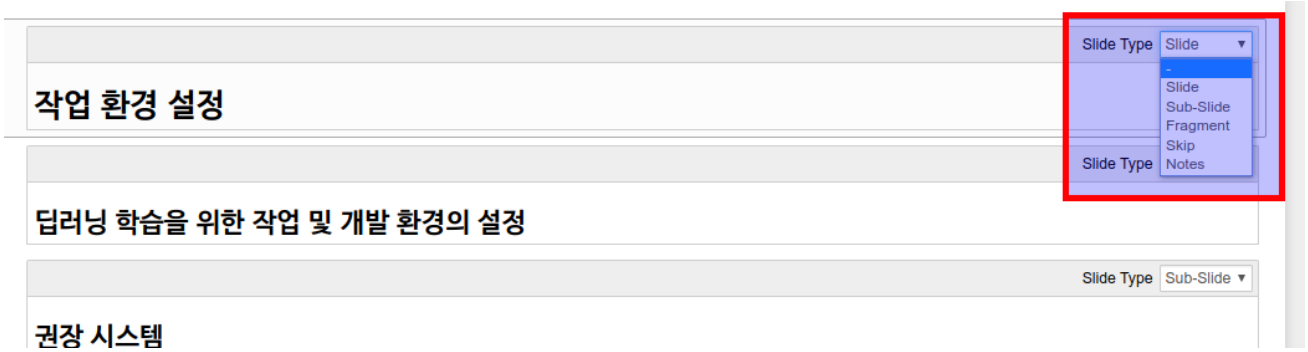
슬라이드로 사용하기 위해서는 먼저 View -> Cell Toolbar -> SlideShow 선택

- 각각의 셀 헤더에 슬라이드 타입 설정 메뉴가 나타남

슬라이드 타입 선택

- 미 지정 혹은 - : 앞 슬라이드에 붙어서 나옴
- `Slide` : 새 슬라이드 화면으로 나옴
- `Sub-Slide` : 서브 슬라이드 화면으로 나옴 - 페이지 전환할 때, 화면이 위로 스크롤 되면서 보임
- `Fragment` : 슬라이드 프라그멘트로 나옴 - 페이지 전환할 때 현재 슬라이드에 추가되어 나옴
- `Skip` : 슬라이드 모드에서는 보이지 않는 셀로 지정
- `Notes` : 발표자 메모 등을 기록. (jupyter notebook 에서는 동작하지 않음)





파이썬 개발 환경 설치 - Docker

- 소개: https://docs.docker.com/engine/installation/linux/docker-ce/ubuntu/ (https://docs.docker.com/engine/installation/linux/docker-ce/ubuntu/)

파이썬 개발 환경 설치 - Docker

```
    sudo apt-get remove docker docker-engine docker.io
    sudo apt-get update
    sudo apt-get install linux-image-extra-$(uname -r) linux-image-extra-virtual
    sudo apt-get install apt-transport-https ca-certificates curl software-
properties-common
    curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
    sudo add-apt-repository "deb [arch=amd64]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
    sudo apt-get update
    sudo apt-get install docker-ce
```

파이썬 개발 환경 설치 - Docker

- `nvidia-docker`
  - https://github.com/NVIDIA/nvidia-docker/releases (https://github.com/NVIDIA/nvidia-docker/releases)
  - 호스트에 설치된 docker 를 이용해서 NVIDIA GPU 를 사용하는 컨테이너 이미지 실행할 때 필요
  - 컨테이너에서 GPU사용을 지원하기 위해서 기본 docker 에서 제공하는 권한 이상의 시스템 권한을 제공
  - 컨테이너에서 실행하는 프로그램이 요구하는 NVIDIA 드라이버 버젼에 맞는 호스트쪽의 드라이버를 자동 관리

파이썬 개발 환경 설치 - Docker

- `nvidia-docker` 설치:

```
# nvidia-docker_1.0.1-1_amd64.deb
sudo dpkg -i nvidia-docker*.deb
```

파이썬 개발 환경 설치 - Docker - airi-tensorflow:latest 설치

```
gzip -d -c airi-tensorflow-py27-2017-08-20.tar.gz | sudo docker load
sudo docker tag airi-tensorflow:py27-2017-08-20 airi-tensorflow:latest
```

파이썬 개발 환경 설치 - Docker

- 사용법(1):

```
nvidia-docker run -ti --rm --pid=host \
    --name my-mnist-model \
    -v /data:/data \
    -v /home:/home \
    -v "$(pwd)":"$(pwd)" \
    -v /etc/localtime:/etc/localtime:ro \
    -w "$(pwd)" \
    airi-tensorflow:latest \
    python
```

파이썬 개발 환경 설치 - Docker

- 사용법(2) - jupyter notebook:

```
nvidia-docker run -ti --rm --pid=host \
    --name my-mnist-model \
    -p 8989:8989 \
    -v /data:/data \
    -v /home:/home \
    -v "$(pwd)":"$(pwd)" \
    -v /etc/localtime:/etc/localtime:ro \
    -w "$(pwd)" \
    airi-tensorflow:latest \
    jupyter notebook --ip=0.0.0.0 --allow-root --no-browser
```

기타

- Google Chrome - https://www.google.com/chrome/ (https://www.google.com/chrome/)
- devdocs.io - http://devdocs.io/ (http://devdocs.io/)
- appear.in - http://appear.in/airi400-class (http://appear.in/airi400-class)
- PyCharm - https://itsfoss.com/install-pycharm-ubuntu/ (https://itsfoss.com/install-pycharm-ubuntu/)

```
sudo add-apt-repository ppa:ubuntu-desktop/ubuntu-make
sudo apt-get update
sudo apt-get install ubuntu-make
umake ide pycharm
```

- Visual Studio Code - https://code.visualstudio.com/ (https://code.visualstudio.com/)
- Atom Editor - https://atom.io/ (https://atom.io/)