파이썬

Python 기초

IPython을 이용한 프로그래밍

2015.03.06 / 이태영

BCcard

- IPython 소개 & 설치
- IPython Notebook 설정과 실행
- IPython Notebook 단축키
- IPython Cell 설명 (Markdown 사용법, Code 셀 특징)
- IPython 매직 명령어
- 운영체제 명령어와 함께 사용
- IPython Notebook 디버깅(ipdb)

파이썬을 배우고 싶어요.

그런데 뭘로 코딩하지?

eclipse

무겁지만 강력하다

IPython

빠르고 강력하다

What is IPython

Interactive computing for python

- Powerful interactive shells (terminal and Qt-based).
- A browser-based notebook with support for code, text, mathematical expressions, inline plots and other rich media.
- Support for interactive data visualization and use of GUI toolkits.
- Flexible, embeddable interpreters to load into your own projects.
- Easy to use, high performance tools for parallel computing.

What is IPython

Interactive computing for python

- 대화형 컴퓨팅으로 분석 프로그래밍 최적합
- 운영체제의 쉘 파일 시스템과 통합되어있음
- 웹기반의 대화형 노트북 지원으로 수식, 표, 그림 등을 표현 가능
- 가볍고 빠른 병렬컴퓨팅 엔진 이용
- 코딩과 문서화, 테스트까지 한 화면에 OK

What is IPython

Interactive computing for python

- python 2.7.9 설치

- numpy 설치
- matplotlib 설치
- pandas 설치
- scipy 설치
- pyzmq 설치
- tornado 설치
- jsonschema 설치
- ipython 설치

- **인터넷이 잘 된다면**

pip install numpy
pip install matplotlib
pip install pandas
pip install scipy
pip install pyzmq
pip install tornado
pip install jsconschema (ipython 3.0 에서 필요)
pip install ipython

- **인터넷이 차단된 환경이라면 (삽질 시작)**

각 라이브러리들의 소스를 구해와서
python setup.py build
python setup.py install

설치 중 특정 라이브러리가 없다며 중단되면
해당 라이브러리 다운로드 받아서 가져와서 설치

What is IPython

Interactive computing for python

```
$ ipython
Python 2.7.9 (default, Feb 13 2015, 23:00:11)
Type "copyright", "credits" or "license" for more information.

IPython 3.0.0 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.

In [1]: import nu <----------------
numbers   numpy

In [1]: import numpy

In [2]: list = [1,2,3,4,5]

In [3]: list
Out[3]: [1, 2, 3, 4, 5]

In [4]: list = [ i for i in range(10) if i%2 != 0 ]

In [5]: list
Out[5]: [1, 3, 5, 7, 9]
```

코드를 적다가 Tab를 누르면
Code assist 기능 제공

What is IPython

Interactive computing for python

```
$ ipython profile create
$ ipython profile create bccard
```

별도로 IPYTHONDIR을 설정하지 않았을 경우 **IPython PATH**

Windows - C:₩Users₩Administrator₩.ipython

Linux - (계정홈)/.ipython

```
./
../
README
extensions/
nbextensions/
profile_bccard/
profile_default/
```

프로필 디렉토리 내의 ipython_notebook_config.py 설정

IPython 2.4.1

```
c = get_config()
c.IPKernelApp.pylab = 'inline'
c.NotebookApp.ip = '*'
c.NotebookApp.open_browser = False
c.NotebookApp.port = 8880
```

IPython 3.0

```
c = get_config()
c.NotebookApp.pylab = 'inline '
c.NotebookApp.ip = '*'
c.NotebookApp.open_browser = False
c.NotebookApp.port = 8880
```

※비밀번호를 세팅하려면 c.NotebookApp.password 항목에 SHA 값을 설정

What is IPython

Interactive computing for python

- **만약 웹서버를 통해 IPytho notebook을 이용할 것이라면 비밀번호 설정 필요**
- 콘솔창에서 ipython 으로 접속하여서 아래처럼 비밀번호 생성

```
In [1]: from IPython.lib import passwd
In [2]: passwd()
Enter password:
Verify password:
Out[2]: 'sha1:3bf58406dd60:a965f3ac8e10a3637c011ba47dec1d081bbaf834'
```

복사 & 붙여넣기

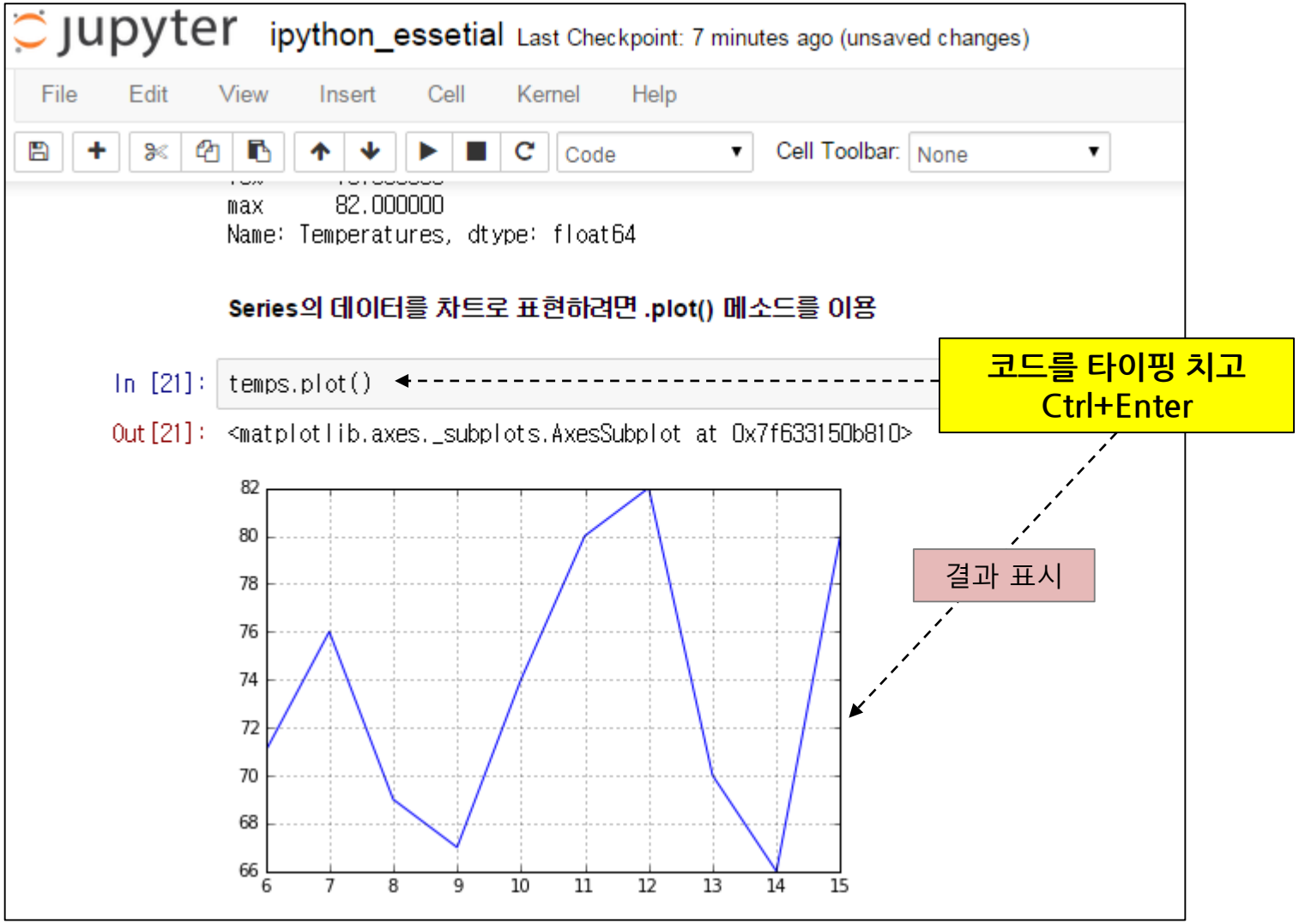
```
c = get_config()
c.IPKernelApp.pylab = 'inline'
c.NotebookApp.ip = '*'
c.NotebookApp.open_browser = False
c.NotebookApp.password = u'sha1:3bf58406dd60:a965f3ac8e10a3637c011ba47dec1d081bbaf834'
c.NotebookApp.port = 9999
```

실행 명령어 ipython notebook --profile=bccard

--profile을 선언하지 않으면 default 프로필로 지정

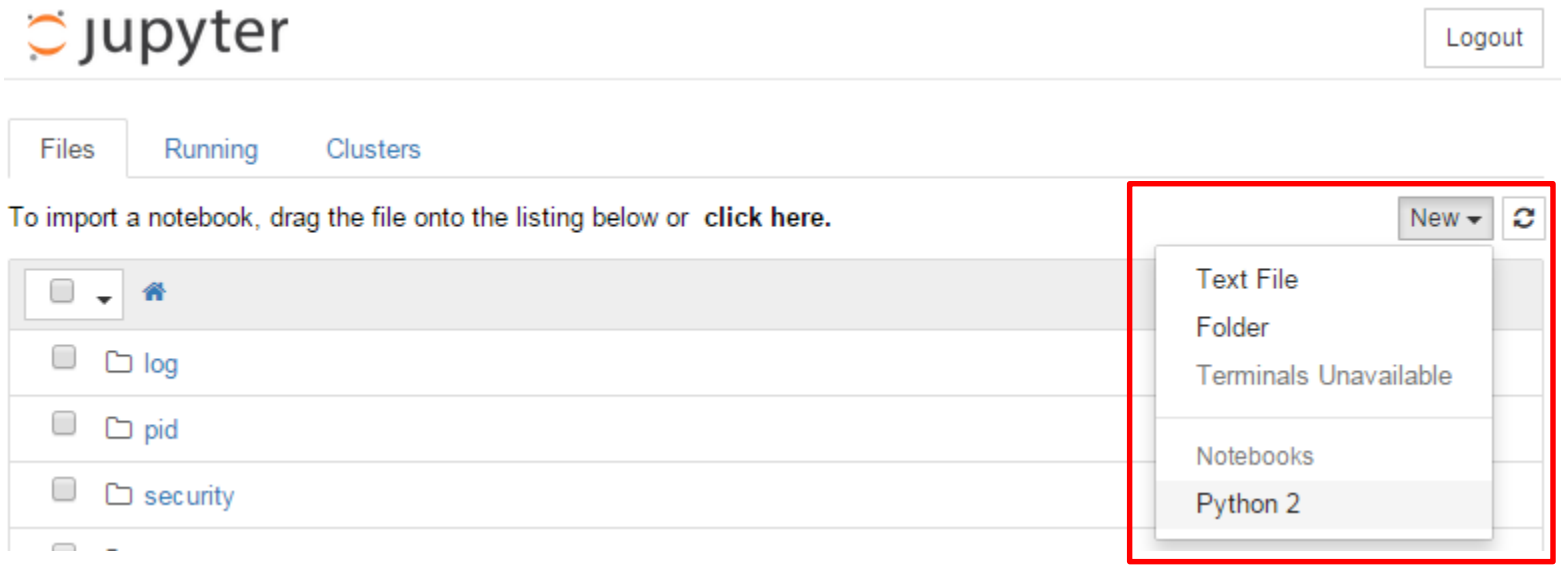
IPython Notebook

Web browser based Python IDE



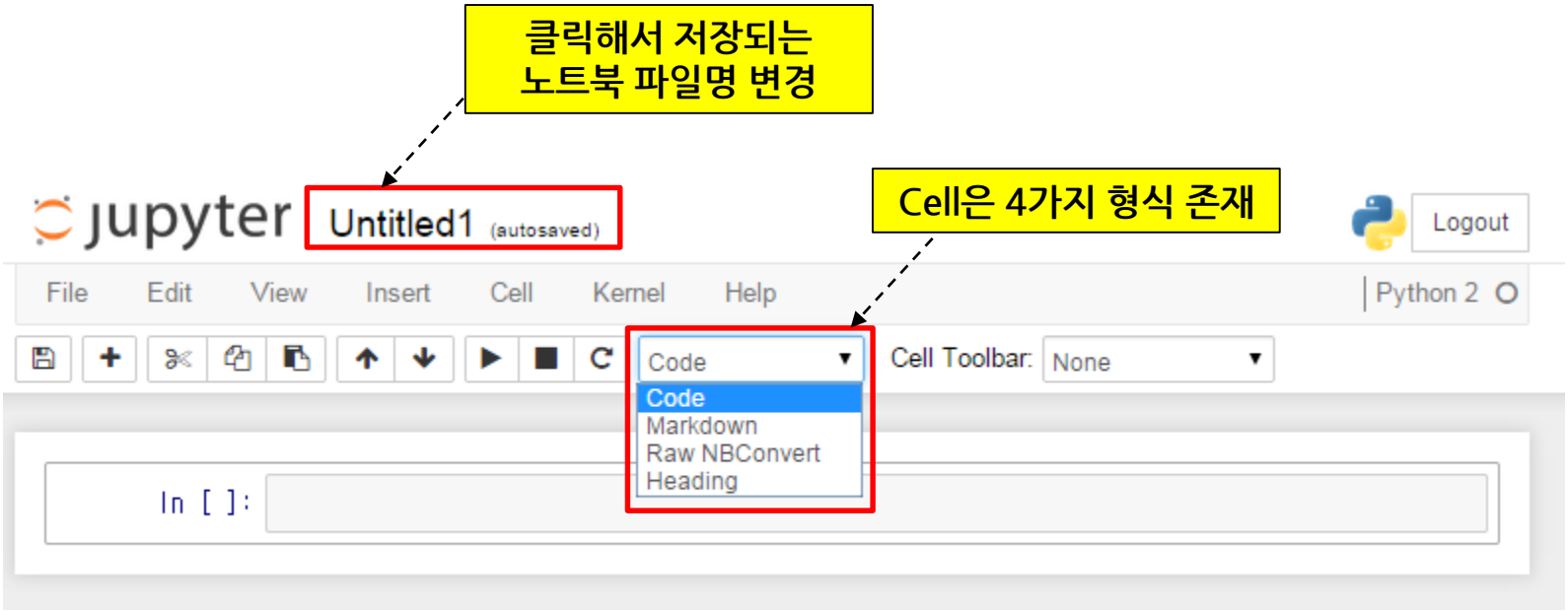
IPython Notebook
파이썬 스크립트 생성

- Notebooks - Python 선택



IPython Notebook
파이썬 스크립트 생성

- Notebooks - Python 선택

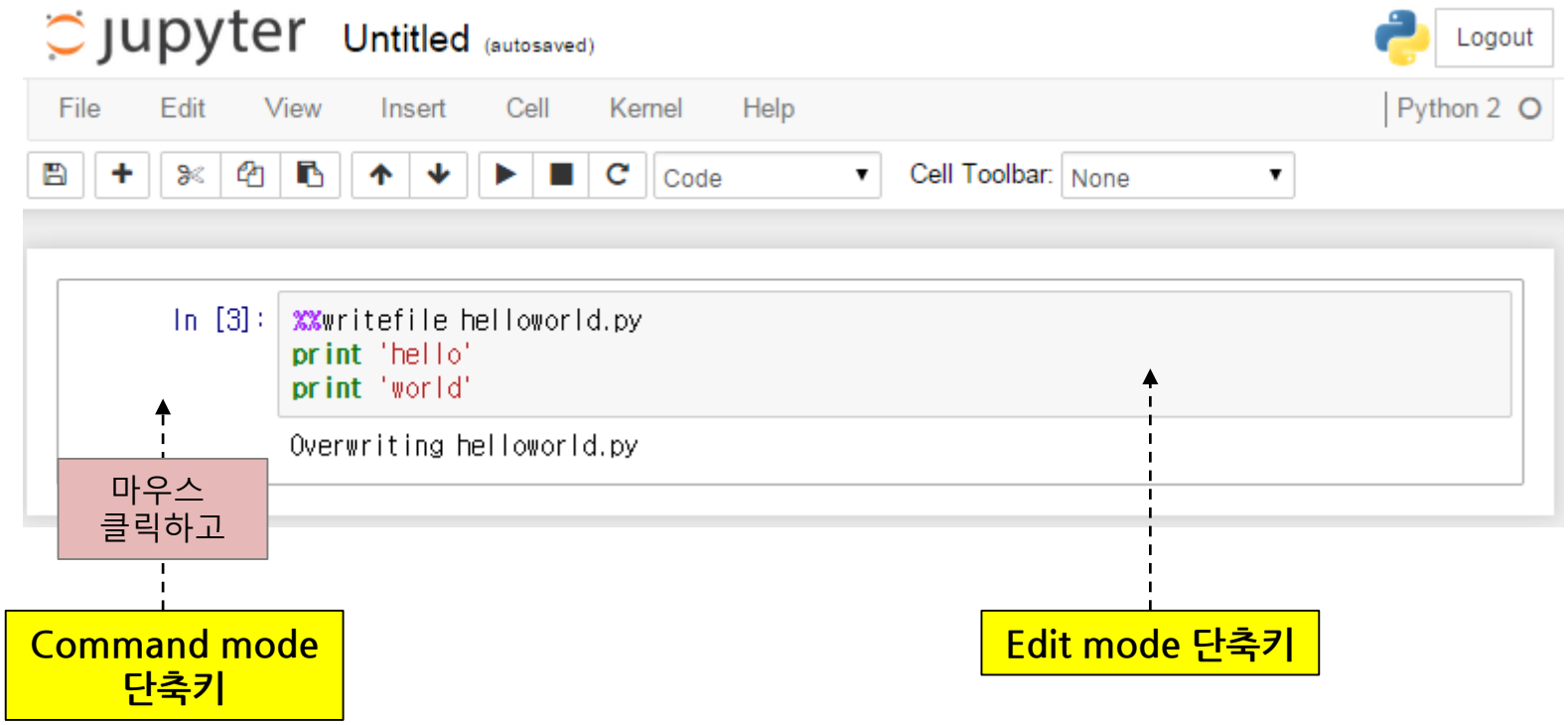


Cell 종류 설명

- Code : 파이썬 코드
- MarkDown : 일종의 마크업 언어. 엔하위키나 텀블러에서 사용 중
- Raw NBCovert : IPython 노트북 날 코딩(?)
- Heading : <html>의 <head> 태그와 유사한 기능

IPython Notebook
단축키 (Shortcut)

- Edit Mode : 셀 안에서 사용하는 단축키
- Command Mode : 셀 경계에서 사용하는 단축키
- h 를 누르면 기본 단축키에 대해 어느 정도 나온다.



IPython Notebook

단축키 (Shortcut) : Edit Mode 단축키

키보드	기능	키보드	기능
Tab	코드 자동완성, 들여쓰기(intent)	Ctrl-Left	한 단어 좌측 이동
Shift-Tab	툴팁	Ctrl-Right	한 단어 우측 이동
Ctrl-]	들여쓰기(intent)	Ctrl-Backspace	이전 단어 삭제
Ctrl-[	내어쓰기(detent)	Ctrl-Delete	이후 단어 삭제
Ctrl-a	전체 선택	Esc	command mode
Ctrl-z	undo	Ctrl-m	command mode
Ctrl-Shift-z	redo	Shift-Enter	run cell, 다음 셀 선택
Ctrl-y	redo	Ctrl-Enter	run cell
Ctrl-Home	셀 처음으로 이동	Alt-Enter	run cell, 다음 셀 삽입
Ctrl-End	셀 끝으로 이동	Ctrl-Shift--	셀 분할
		Ctrl-s	노트북 파일 저장

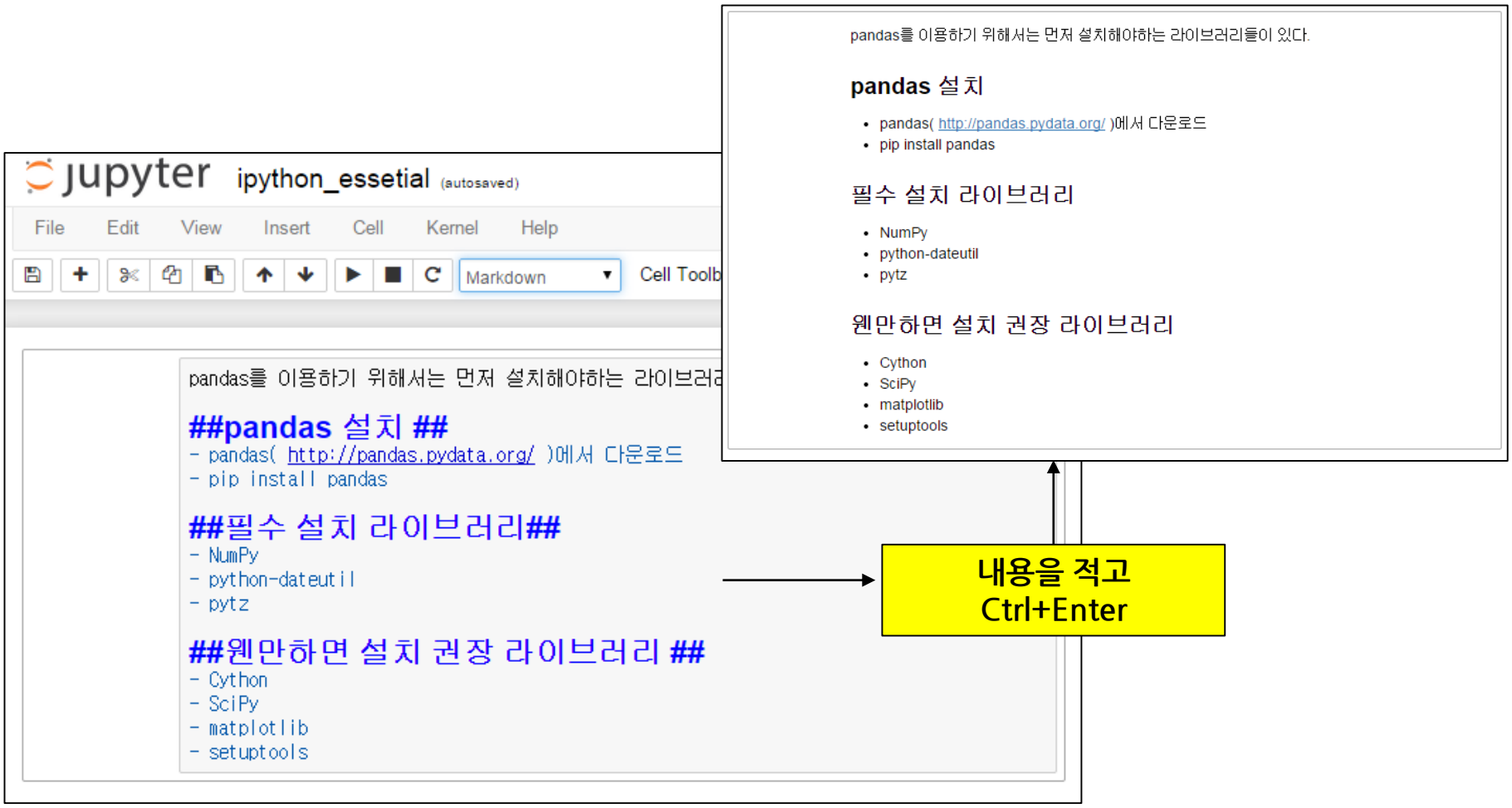
IPython Notebook

단축키 (Shortcut) : Command Mode 단축키

키보드	기능	키보드	기능
Enter	Edit mode 진입	c	셀 복사
Shift-Enter	run cell, 다음 셀 선택	Shift-v	위에 붙여넣기
Ctrl-Enter	run cell	v	아래 붙여넣기
Alt-Enter	run cell, 다음 셀 삽입	x	셀 잘라내기
y	Code 셀로 변환	dd	셀 삭제
m	Markdown 셀로 변환	z	undo 이전 삭제 복구
r	Raw NB 셀로 변환	Shift-m	아래 셀과 병합(merge)
1	Markdown : H1	s	노트북 저장
2	Markdown : H2	Ctrl-s	노트북 저장
3	Markdown : H3	l	해당 셀의 라인번호 토글
4	Markdown : H4	o	결과값 토글
5	Markdown : H5	Shift-o	결과값 토글(scrolling)
6	Markdown : H6	h	keyboard shortcuts
a	위에 셀 삽입	ii	interrupt kernel
b	아래 셀 삽입	00	restart kernel
k / 방향키 위	이전 셀로 이동	j / 방향키 아래	다음 셀로 이동

IPython Notebook
Cell : Markdown

- Markdown language 기반으로 Document 작성 (Wikipedia처럼 작성 방식)
- 코드에 대한 설명이 가능하다.
- 수학수식 표현도 가능 (Mathjax)



IPython Notebook
Cell : Markdown

- **제목(Head 태그)**

  # 텍스트

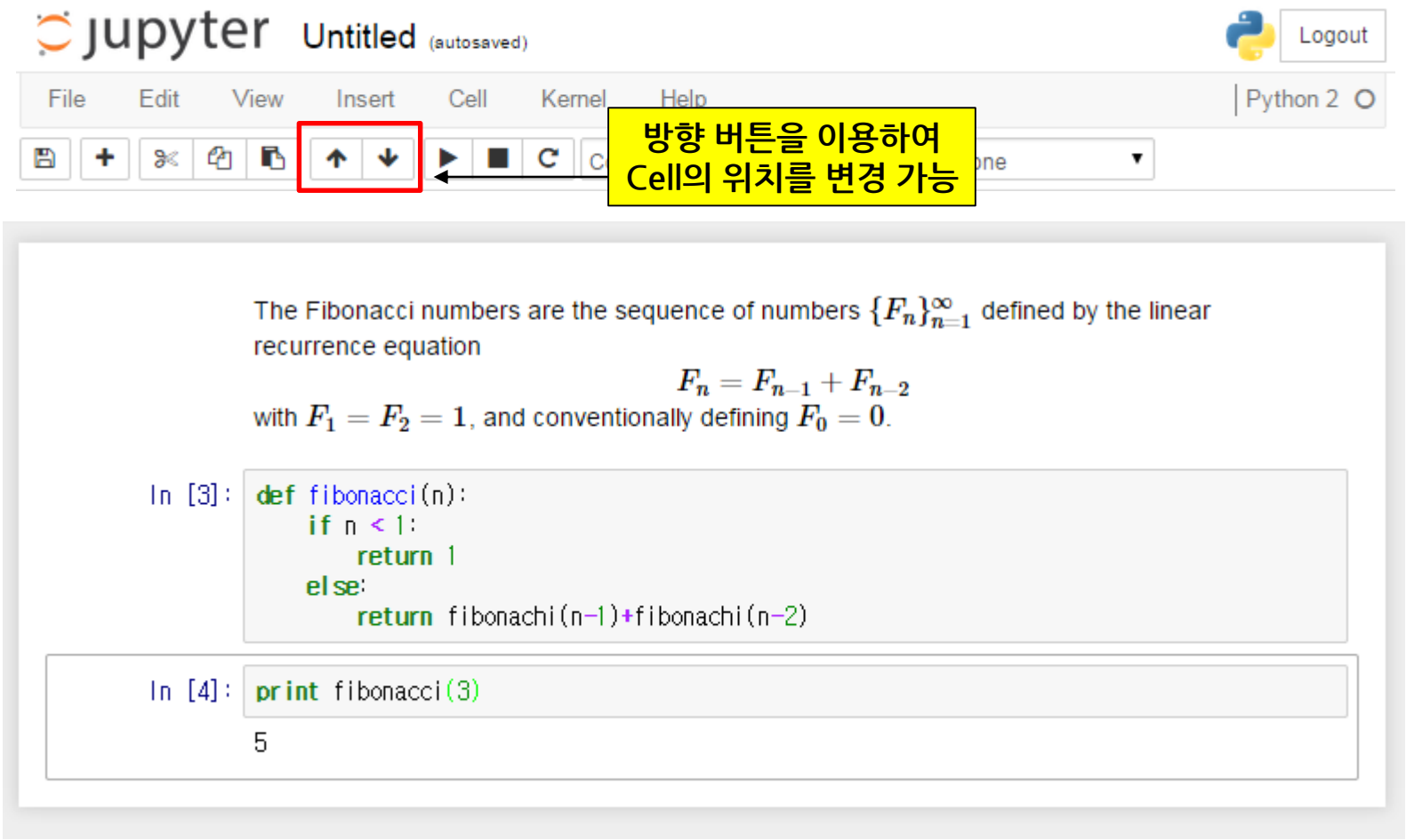
  # 하나를 쓰면 HTML의 <h1> 태그

  # 두개를 쓰면 HTML의 <h2> 태그. 최대 6개까지 쓸 수 있다.

- **리스트 (li 태그)**
  - 번호 없는 리스트 : -텍스트 , +텍스트, *텍스트
  - 번호 있는 리스트 : 숫자.텍스트
- **기울인 글씨(Italic 태그)** : _텍스트_ , *텍스트*
- **굵은 글씨(Strong 태그) :** __텍스트__ , **텍스트**
- **인용(cite)** : >텍스트
- **인라인 링크** : [텍스트](링크)
- **수평선(hr 태그) :** --- , *** , ___

IPython Notebook
Cell : Code

- 파이썬의 코드를 각 셀에 원하는 만큼을 작성하여 실행하면 메모리 상에 반영된다.
- 위, 아래 위치가 달라도 실행한 스텝 번호(In [번호])가 높을 수록 최근에 수행된 영역이다.



IPython Notebook
Magic Keyword (매직 명령어)

- **자주 사용하는 IPython 매직 명령어**
  - %(명령어) : 셀 내의 라인 독립 실행
  - %%(명령어) : 셀 내의 전체 내용 관련 실행

매직 명령어	기능
%quickref	IPython의 빠른 도움말 표시
%magic	모든 매직 함수에 대한 상세 도움말 출력
%debug	최근 예외 트레이스백의 하단에서 대화형 디버거로 진입
%hist	명령어 입력(그리고 선택적 출력) history 출력
%pdb	예외가 발생하면 자동으로 디버거로 진입
%run script.py	IPython 내에서 파이썬 스크립트 실행
%timeit statement	statement를 여러차례 실행한 후 평균 실행 시간을 출력. 매우 짧은 시간 안에 끝나는 코드의 시간을 측정할 때 유용
%%writefile 파일명	filename인 파일을 생성
%save 파일명 from-to	IPython의 이미 실행된 Step번호 구간대 명령어를 파일로 저장

IPython Notebook
Magic Keyword (매직 명령어)

- 자주 사용하는 IPython 매직 명령어
  - %(명령어) : 셀 내의 라인 독립 실행
  - %%(명령어) : 셀 내의 전체 내용 관련 실행

In [1]:
```
%pdb
```
```
Automatic pdb calling has been turned ON
```

In [6]:
```
%%writefile fibonacci.py
def fibonacci(n):
    if n < 1:
        return 1
    else:
        return fibonacci(n-1)+fibonacci(n-2)
```
```
Writing fibonacci.py
```

In [10]:
```
from fibonacci import *
```

In [11]:
```
fibonacci(10)
```

Out[11]:
```
144
```

In [4]:
```
%timeit fibonacci(10)
```
```
10000 loops, best of 3: 123 µs per loop
```

IPython Notebook

Magic Keyword (매직 명령어) : TIP

- **(리눅스/유닉스만) %%writefile으로 스크립트 저장 후 %run으로 실행하려면**

  파이썬 스크립트에 한글이 포함될 때는 #!(샤방) 파이썬 패스, 언어 타입 정의 필요하다.

  ```
  #!/usr/bin/python
  #-*- coding: utf-8
  ```

In [13]:
```
%%writefile fibonacci.py
#!/usr/bin/python
#-*-coding:utf-8
#피보나치 수열 함수
def fibonacci(n):
    if n < 1:
        return 1
    else:
        return fibonacci(n-1)+fibonacci(n-2)

#피보나치 수열 실행부 출력
print '피보나치 f(10)=', fibonacci(10)
```

```
Overwriting fibonacci.py
```

In [14]:
```
%run fibonacci.py
```

```
피보나치 f(10)= 144
```

IPython Notebook
운영체제와 함께 사용하기

- IPython에서 운영체제의 자체 명령어를 실행하기 위해서는 !(명령어)로 수행

```
In [1]: !pwd
/home/seen/.ipython/profile_2025

In [2]: !ls
2nd.ipynb           fibonacci.pyc              nohup.out
3rd.ipynb           helloworld.py              pandas.ipynb
Untitled.ipynb      ipython_config.py          pid
Untitled1.ipynb     ipython_console_config.py  security
Untitled2.ipynb     ipython_kernel_config.py   startup
cluster.ipynb       ipython_notebook_config.py static
cluster_run.ipynb   log                        untitled.txt
fibonacci.py        matplotlib.ipynb

In [3]: !tail nohup.out
        u'output_type': u'stream',
        u'stream': u'stdout',
        u'text': u'File `cluster.py` exists. Overwrite (y/[N])?  y₩n'}
[W 00:06:22.542 NotebookApp] Notebook cluster.ipynb is not trusted
[I 00:08:32.918 NotebookApp] Saving file at /Untitled.ipynb
[I 00:08:58.384 NotebookApp] Saving file at /Untitled.ipynb
[I 00:09:19.128 NotebookApp] Creating new notebook in
[I 00:09:19.657 NotebookApp] Kernel started: 7f0b4ee8-e376-44ff-a48a-448bd38dddf2
[I 00:09:39.630 NotebookApp] Saving file at /Untitled2.ipynb
[I 00:10:52.405 NotebookApp] Kernel restarted: 7f0b4ee8-e376-44ff-a48a-448bd38dddf2
```

IPython Notebook
운영체제와 함께 사용하기

- !(명령어)로 수행된 출력결과를 IPython의 데이터로 사용 가능

In [2]:
```
!ls
```

```
2nd.ipynb           fibonacci.pyc               nohup.out
3rd.ipynb           helloworld.py               pandas.ipynb
Untitled.ipynb      ipython_config.py           pid
Untitled1.ipynb     ipython_console_config.py   security
Untitled2.ipynb     ipython_kernel_config.py    startup
cluster.ipynb       ipython_notebook_config.py  static
cluster_run.ipynb   log                         untitled.txt
fibonacci.py        matplotlib.ipynb
```

In [4]:
```
list = !ls
```

In [9]:
```
for file in list:
    print file
```

```
2nd.ipynb
3rd.ipynb
Untitled.ipynb
Untitled1.ipynb
Untitled2.ipynb
cluster.ipynb
cluster_run.ipynb
fibonacci.py
fibonacci.pyc
helloworld.py
ipython_config.py
ipython_console_config.py
ipython_kernel_config.py
ipython_notebook_config.py
log
```

IPython Notebook
ipdb 디버깅을 이용한 코드 분석

- 매직 명령어 %debug 혹은 %%debug

In [*]:
```
%%debug
#피보나치 수열 함수
def fibonacci(n):
    if n < 1:
        return 1
    else:
        return fibonacci(n-1)+fibonacci(n-2)

#피보나치 수열 실행부 출력
print '피보나치 f(10)=', fibonacci(10)
```

```
NOTE: Enter 'c' at the ipdb>  prompt to continue execution.
None
> <string>(3)<module>()

ipdb>
```

In [*]:
```
%debug fibonacci(10)
```

```
NOTE: Enter 'c' at the ipdb>  prompt to continue execution.
> <string>(1)<module>()

ipdb>
```

IPython Notebook

ipdb 디버깅을 이용한 코드 분석

- ipdb 명령어

명령어		설명
h	help	디버그 도움말. h만 입력하면 디버그 명령어 리스트 출력 h [명령어]를 입력하면 [명령어]에 대한 이용 도움말
w	where	현재 위치의 Stack trace 출력
s	step	현재 라인을 실행하고 다음 스텝으로 이동
n	next	현재 라인을 실행하고 다음 라인으로 이동
r	return	현재 함수가 끝날 때(return)까지 계속 실행
b	break	특정 라인에 break point 설정
c	continue	break point가 있을 때까지 계속 실행
a	args	현재 함수에 할당된 argument를 출력
p	print	value를 출력
cl	clear [args]	clear break point
d	down	현재 Stack에서 하위 Strack frame으로 이동
u	up	현재 Stack에서 상위 Stack frame으로 이동
l	list [from,to]	소스 출력
run	run [args]	Restart 디버거 프로그램. [args]을 입력하면 sys.argv에 할당됨
q	quit	디버깅 종료

IPython Notebook

ipdb 디버깅을 이용한 코드 분석

- h (help) 명령어

```
NOTE: Enter 'c' at the ipdb>  prompt to continue execution.
> <string>(1)<module>()

ipdb> h

Documented commands (type help <topic>):
========================================
EOF    bt         cont      enable  jump  pdef    psource  run      unt
a      c          continue  exit    l     pdoc    q        s        until
alias  cl         d         h       list  pfile   quit     step     up
args   clear      debug     help    n     pinfo   r        tbreak   w
b      commands   disable   ignore  next  pinfo2  restart  u        whatis
break  condition  down      j       p     pp      return   unalias  where

Miscellaneous help topics:
==========================
exec  pdb

Undocumented commands:
======================
retval  rv

ipdb> h c
c(ont(inue))
Continue execution, only stop when a breakpoint is encountered.
ipdb> h b
b(reak) ([file:]lineno | function) [, condition]
With a line number argument, set a break there in the current
file.  With a function name, set a break at first executable line
of that function.  Without argument, list all breaks.  If a second
argument is present, it is a string specifying an expression
which must evaluate to true before the breakpoint is honored.
```

IPython Notebook

ipdb 디버깅을 이용한 코드 분석

- step , up, down 명령어

```
In [8]:
def A():
    print 'world'
def B():
    print 'hello'
    A()
```

```
%debug B()
```

다음 스텝

상위 Stack으로 이동

하위 Stack으로 이동

```
ipdb> b 5
Breakpoint 1 at <ipython-input-8-c7a559c3b1ab>:5
ipdb> c
hello
> <ipython-input-8-c7a559c3b1ab>(5)B()
      3 def B():
      4     print 'hello'
1---> 5     A()

ipdb> s
--Call--
> <ipython-input-8-c7a559c3b1ab>(1)A()
----> 1 def A():
      2     print 'world'
      3 def B():

ipdb> s
> <ipython-input-8-c7a559c3b1ab>(2)A()
      1 def A():
----> 2     print 'world'
      3 def B():

ipdb> u
> <ipython-input-8-c7a559c3b1ab>(5)B()
      3 def B():
      4     print 'hello'
1---> 5     A()

ipdb> d
> <ipython-input-8-c7a559c3b1ab>(2)A()
      1 def A():
----> 2     print 'world'
      3 def B():
```

A 함수로 진입

Q&A