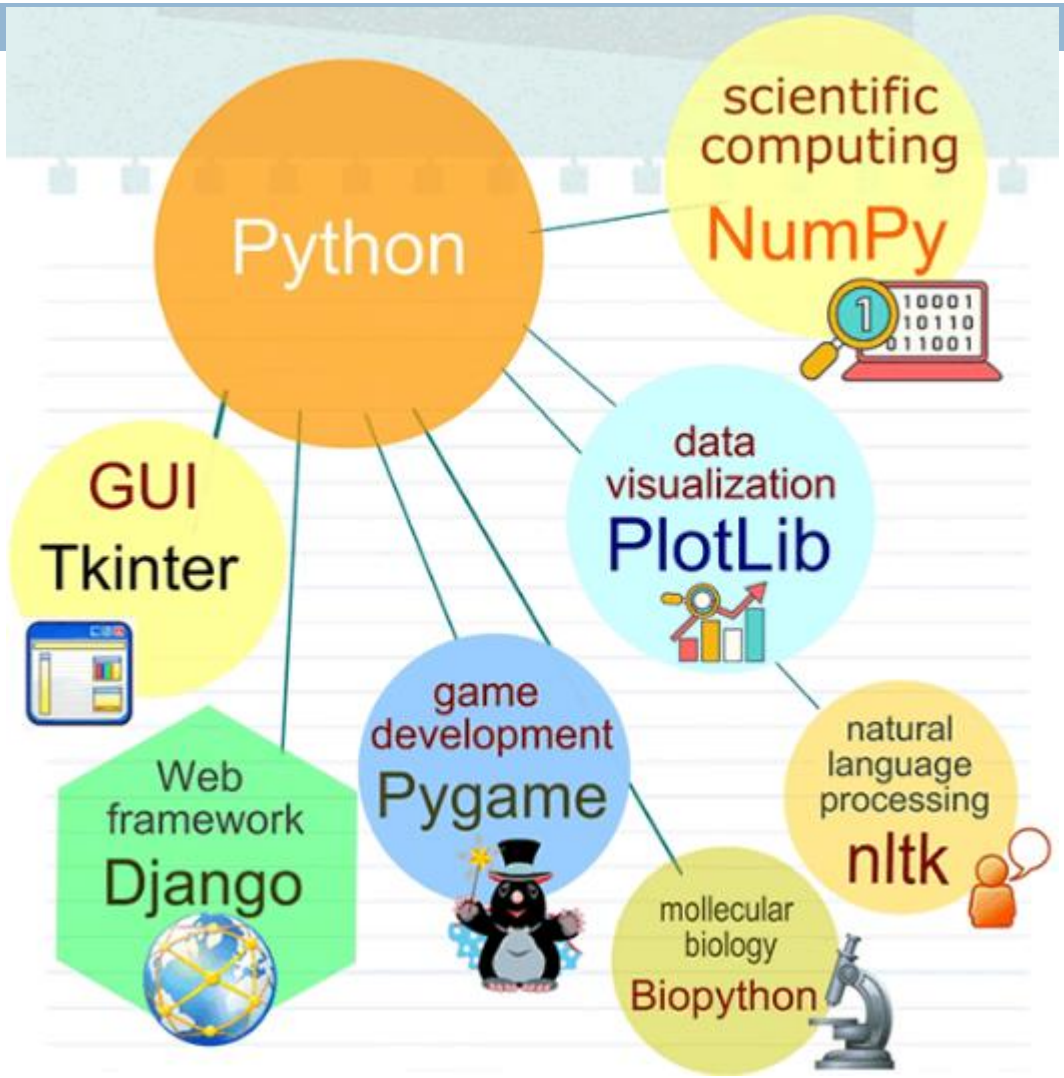
12장 다양한 라이브러리를 사용하자

파이썬의 외부 라이브러리

- PIL(Python Imaging Library) 또는 필로우(Pillow)
  - 파이썬에서 영상 처리 라이브러리를 찾는다면 누구나 PIL(Python Imaging Library)이라고 할 것이다. 하지만 최근에는 업데이트가 잦지 않다는 단점이 있다. 필로우는 PIL와 호환성을 유지하면서 쉽게 사용할 수 있도록 한 라이브러리이다. 파이썬의 Tkinter 모듈과의 호환성도 가지고 있다.
- 파이게임(Pygame)
  - 파이썬으로 게임을 제작하기 위한 프레임워크이다. 파이게임(Pygame)은 캔버스와 그래픽 그리기, 다채널 사운드 처리, 클릭 이벤트 처리, 충돌 감지 등의 작업을 지원한다.
- 넘파이(NumPy)
  - 넘파이는 통계, 선형 대수, 행렬 계산, 금융 운용 등을 포함한 과학 계산과 수학 작업에 많이 사용되는 라이브러리이다. 금융 시장 분석가나 회계 담당자는 넘파이(NumPy)을 사용하면 좋다.
- 사이파이(SciPy)
  - 거의 완벽한 파이썬 '과학-수학' 기능을 지원한다.

파이썬의 라이브러리


Python
scientific computing
NumPy
10001
10110
11001
GUI
Tkinter
data visualization
PlotLib
game development
Pygame
Web framework
Django
natural language processing
nltk
mollecular biology
Biopython

필로우 설치

- 파이썬 패키지를 설치할 때 가장 많이 사용하는 도구가 바로 pip이다

```
d:\>pip

Usage:
  pip <command> [options]

Commands:
  install                     Install packages.
  download                    Download packages.
 uninstall                    Uninstall packages.
  freeze                      Output installed packages in
requirements format.
...
d:\>
```

필로우 설치

- 이제 pip를 이용하여 필로우 라이브러리를 설치해보자.

```
d:\>pip install Pillow
Collecting Pillow
  Downloading Pillow-3.3.0-cp35-cp35m-win32.whl (1.3MB)
    100% |################################| 1.3MB 867kB/s
Installing collected packages: Pillow
Successfully installed Pillow-3.3.0
You are using pip version 8.1.1, however version 8.1.2 is
available.
You should consider upgrading via the 'python -m pip install --
upgrade pip' comm
and.

d:\>
```

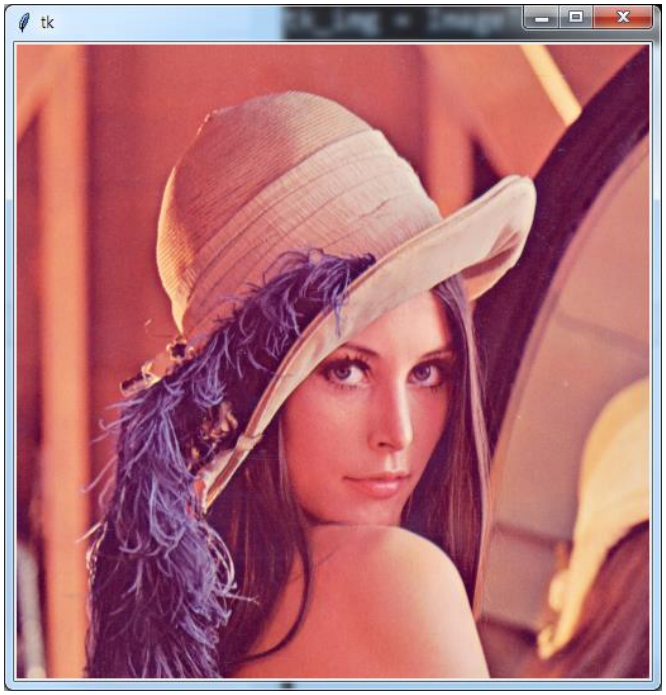
pip 버전 업그레이드

- pip 버전 업그레이드

```
D:\>python -m pip install --upgrade pip
Collecting pip
  Using cached pip-8.1.2-py2.py3-none-any.whl
Installing collected packages: pip
  Found existing installation: pip 7.1.2
    Uninstalling pip-7.1.2:
      Successfully uninstalled pip-7.1.2
Successfully installed pip-8.1.2
```

필로우를 이용한 영상 표시

- 필로우는 많은 영상 포맷을 지원한다. BMP, EPS, GIF, IM, JPEG, MSP, PCX, PNG, PPM, TIFF, WebP, ICO, PSD, PDF 등의 형식을 지원한다.

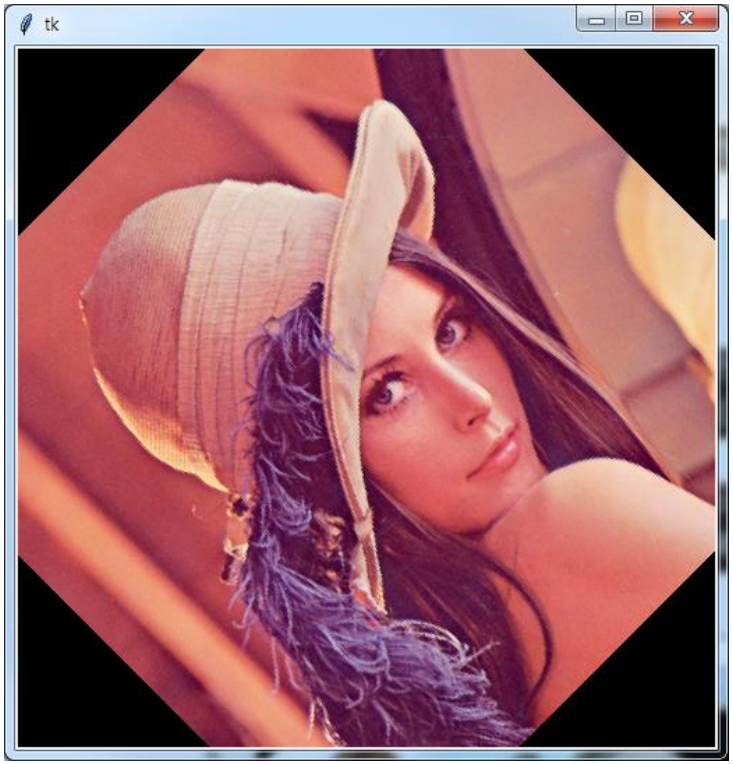


예제

```
# PIL 모듈에서 몇 개의 클래스를 포함시킨다.
from PIL import Image, ImageTk

# tkinter 모듈을 포함시킨다.
import tkinter as tk

# 윈도우를 생성하고 윈도우 안에 캔버스를 생성한다.
window = tk.Tk()
canvas = tk.Canvas(window, width=500, height=500)
canvas.pack()

# 윈도우를 생성하고 윈도우 안에 캔버스를 생성한다.
img = Image.open("D:\\lenna.png")

# tk 형식으로 영상을 변환한다.
tk_img = ImageTk.PhotoImage(img)

# tkinter의 캔버스에 영상을 표시한다.
canvas.create_image(250, 250, image=tk_img)

window.mainloop()
```

필로우를 이용한 영상 처리

- 이번에는 영상을 읽어서 45도 회전한 후에 화면에 표시해본다.

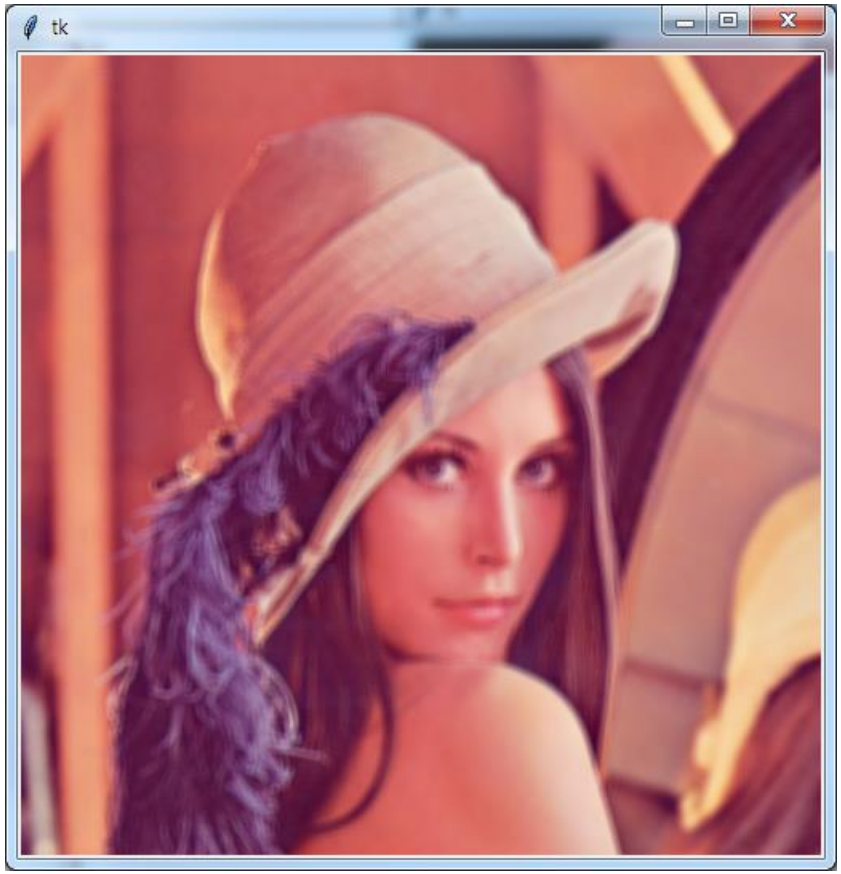


예제

```
from PIL import Image, ImageTk
import tkinter as tk

window = tk.Tk()
canvas = tk.Canvas(window, width=500, height=500)
canvas.pack()

# 영상 파일을 연다.
im = Image.open("d:\\lenna.png")

# 영상을 45도 회전한다.
out = im.rotate(45)

# 영상을 tkinter 형식으로 변환한다.
tk_img = ImageTk.PhotoImage(out)

# 영상을 tkinter에서 화면에 표시한다.
canvas.create_image(250, 250, image=tk_img)
window.mainloop()
```

영상 흐리게 하기

- 이번에는 영상을 읽어서 흐리게 한 후에 화면에 표시해 본다.

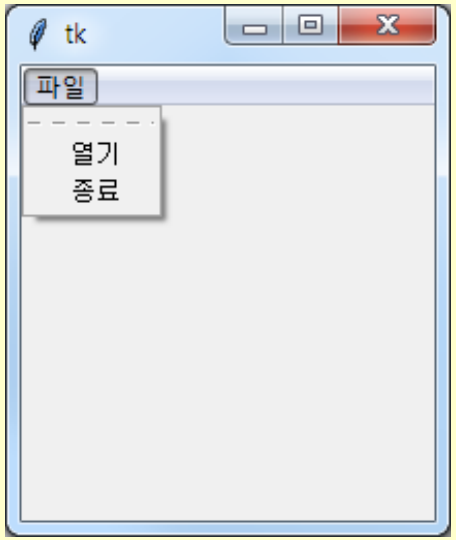


예제

```
from PIL import Image, ImageTk, ImageFilter
import tkinter as tk

window = tk.Tk()
canvas = tk.Canvas(window, width=500, height=500)
canvas.pack()

# 영상 파일을 연다.
im = Image.open("d:\\lenna.png")

# 영상을 흐리게 한다.
out = im.filter(ImageFilter.BLUR)

# 영상을 tkinter 형식으로 변환한다.
tk_img = ImageTk.PhotoImage(out)

# 영상을 tkinter에서 화면에 표시한다.
canvas.create_image(250, 250, image=tk_img)
window.mainloop()
```

메뉴 만들기

```
import tkinter as tk

def open():
    pass

def quit():
    window.quit()

window = tk.Tk()
menubar = tk.Menu(window)
filemenu = tk.Menu(menubar)

filemenu.add_command(label="열기", command=open)
filemenu.add_command(label="종료", command=quit)

menubar.add_cascade(label="파일", menu=filemenu)

window.config(menu=menubar)
window.mainloop()
```



영상 처리 기능을 메뉴로 연결

```
from PIL import Image, ImageTk, ImageFilter
import tkinter as tk
from tkinter import filedialog as fd

im = None
tk_img = None

# 파일 메뉴에서 “열기”를 선택하였을 때 호출되는 함수
def open():
    global im, tk_img
    fname = fd.askopenfilename()
    im = Image.open(fname)
    tk_img = ImageTk.PhotoImage(im)
    canvas.create_image(250, 250, image=tk_img)
    window.update()

# 파일 메뉴에서 “종료”를 선택하였을 때 호출되는 함수
def quit():
    window.quit()
```

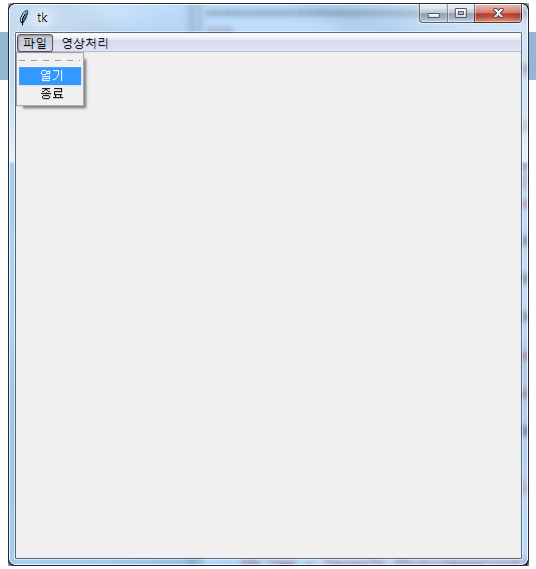
영상 처리 기능을 메뉴로 연결

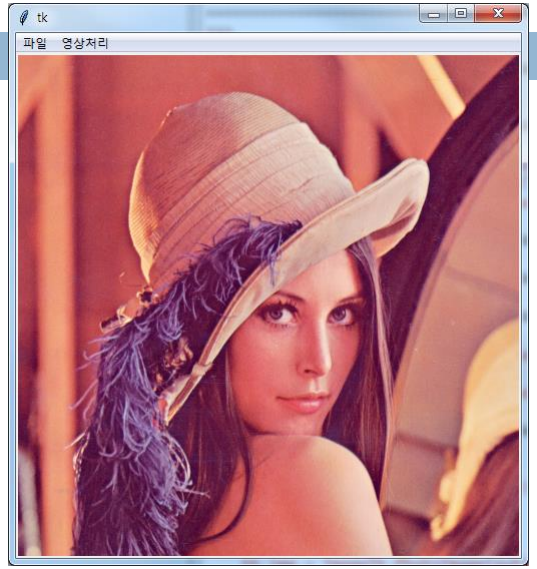
```
# 영상처리 메뉴에서 “열기”를 선택하였을 때 호출되는 함수
def image_rotate():
    global im, tk_img
    out = im.rotate(45)
    tk_img = ImageTk.PhotoImage(out)
    canvas.create_image(250, 250, image=tk_img)
    window.update()

# 영상처리 메뉴에서 “열기”를 선택하였을 때 호출되는 함수
def image_blur():
    global im, tk_img
    out = im.filter(ImageFilter.BLUR)
    tk_img = ImageTk.PhotoImage(out)
    canvas.create_image(250, 250, image=tk_img)
    window.update()

# 윈도우를 생성한다.
window = tk.Tk()
canvas = tk.Canvas(window, width=500, height=500)
canvas.pack()
```

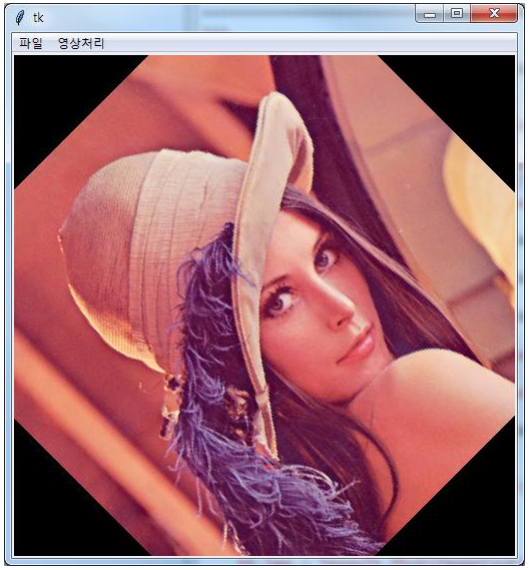
영상 처리 기능을 메뉴로 연결

```
# 메뉴를 생성한다.
menubar = tk.Menu(window)
filemenu = tk.Menu(menubar)
ipmenu = tk.Menu(menubar)
filemenu.add_command(label="열기", command=open)
filemenu.add_command(label="종료", command=quit)
ipmenu.add_command(label="영상회전", command=image_rotate)
ipmenu.add_command(label="영상흐리게", command=image_blur)
menubar.add_cascade(label="파일", menu=filemenu)
menubar.add_cascade(label="영상처리", menu=ipmenu)

window.config(menu=menubar)
window.mainloop()
```

실행 결과







이번 장에서 배운 것

- *pip를 사용하여 외부 라이브러리르 설치해보았다.*
- *필로우 라이브러리를 이용하여 영상을 처리해보았다.*
- *메뉴를 사용해보았다.*

Q & A