Chapter

4

OpenCV 인터페이스

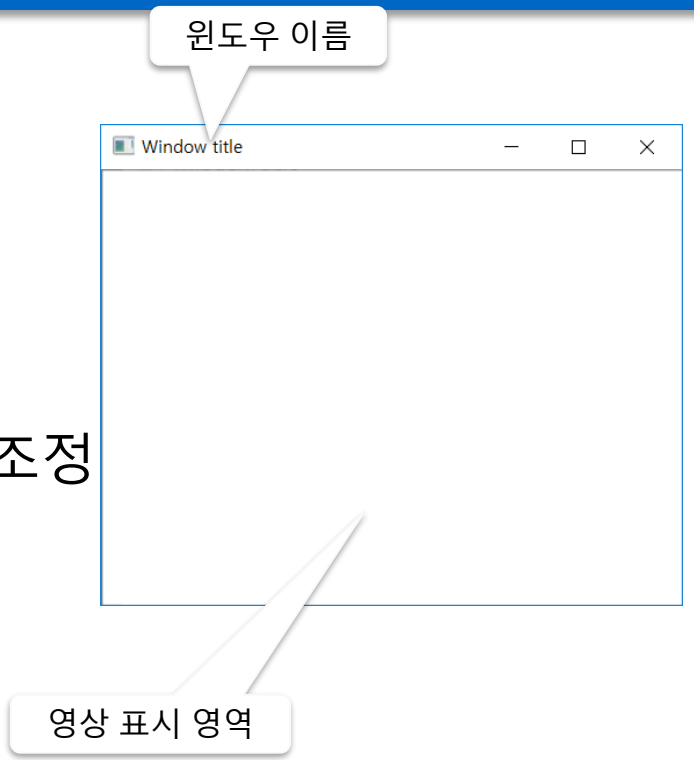
1

윈도우 제어

윈도우 제어

❖ cv2.namedWindow(winname[, flag]) → None

- 윈도우 이름을 설정한 후, 해당 이름으로 윈도우 생성
- flag
  - cv2.WINDOW_NORMAL : 윈도우 크기 재조정 가능
  - cv2.WINDOW_AUTOSIZE : 표시될 행렬의 크기에 맞춰 자동 조정



❖ cv2.imshow(winname, mat) → None

- winname 윈도우에 mat 행렬을 영상으로 표시
- 생성된 윈도우가 없으면, winname 이름으로 윈도우를 생성
- mat : numpy의 ndarray

❖ **cv2.destroyAllWindows() → None**

- HighGUI로 생성된 모든 윈도우 소멸

❖ **cv2.destroyWindow(winname) → None**

- winname으로 지정된 윈도우 소멸

❖ **cv2.moveWindow(winname, x, y) → None**

- winname 윈도우를 (x, y) 위치로 이동. 원점은 모니터의 좌측상단.

❖ **cv2.resizeWindow(winname, width, height) → None**

- winname 윈도우의 크기를 재조정

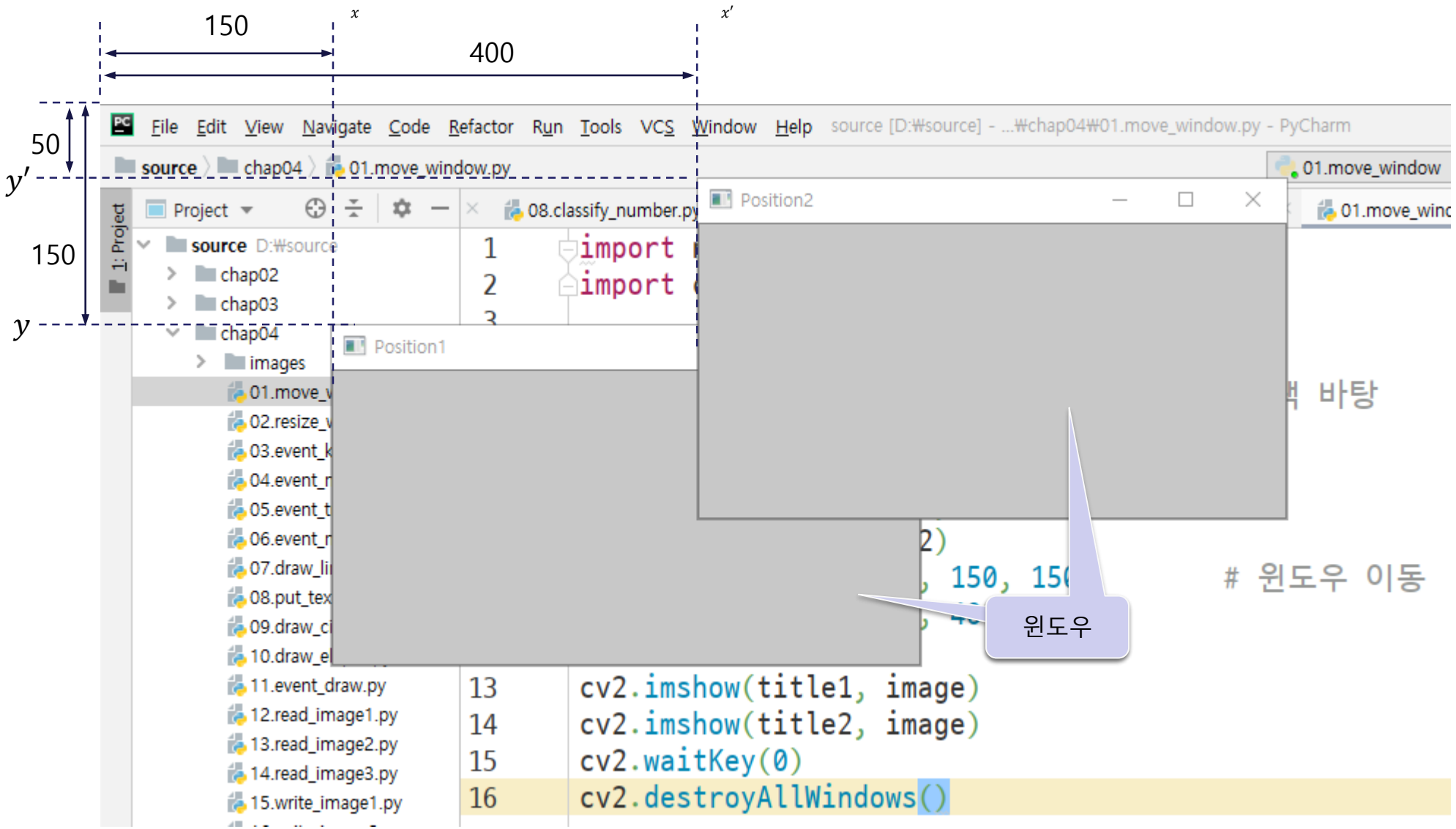
라이브러리 임포트

0 원소 행렬 생성

슬라이스 연산자로 행렬 원소값 지정

예제 4.1.1 윈도우 이동 – 01.move_window.py

```
01  import numpy as np                          # 넘파이 라이브러리 임포트
02  import cv2                                  # OpenCV 라리브러리 임포트
03
04  image = np.zeros((200, 400), np.uint8)      # 행렬 생성
05  image[:] = 200                              # 밝은 회색(200) 바탕 영상 생성
06
07  title1, title2 = 'Position1', 'Position2'   # 윈도우 이름
08  cv2.namedWindow(title1, cv2.WINDOW_AUTOSIZE) # 윈도우 생성 및 크기 조정 옵션
09  cv2.namedWindow(title2)
10  cv2.moveWindow(title1, 150, 150)            # 윈도우 이동 – 위치 지정
11  cv2.moveWindow(title2, 400, 50)
12
13  cv2.imshow(title1, image)                   # 행렬 원소를 영상으로 표시
14  cv2.imshow(title2, image)
15  cv2.waitKey(0)                              # 키 이벤트(key event) 대기
16  cv2.destroyAllWindows()                     # 열린 모든 윈도우 파괴
```

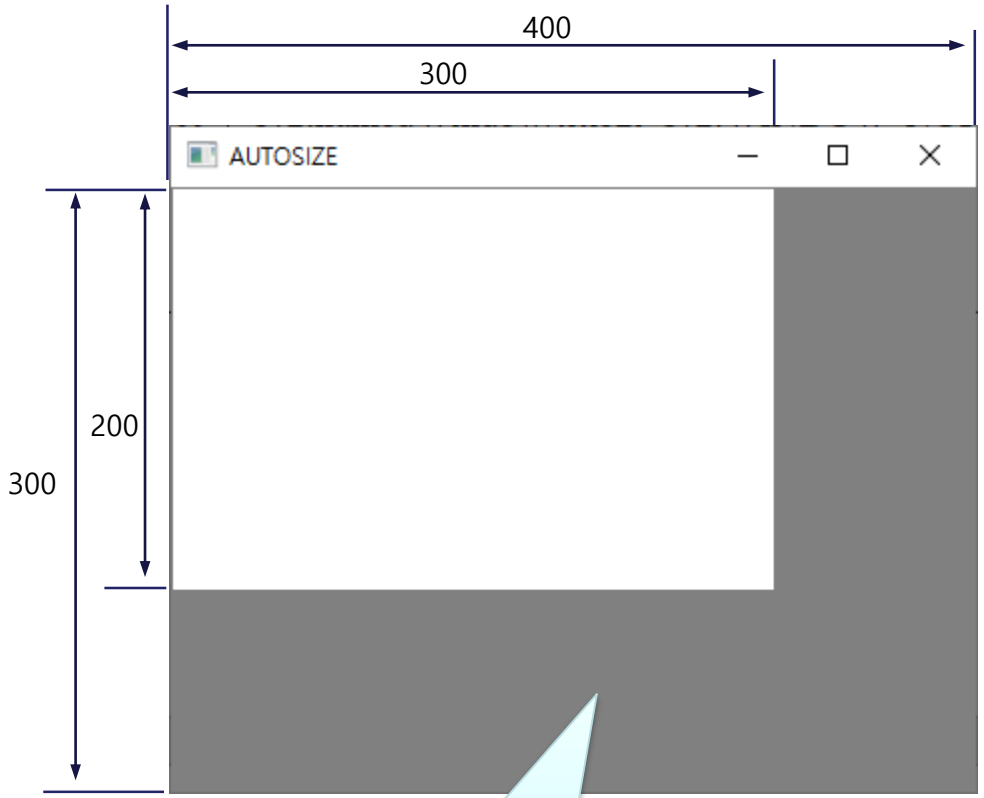

150
400
x
x′
50
y′
150
y
File Edit View Navigate Code Refactor Run Tools VCS Window Help
source [D:₩source] - ...₩chap04₩01.move_window.py - PyCharm
source chap04 01.move_window.py
01.move_window
Project
08.classify_number.py
source D:₩source
chap02
chap03
chap04
images
01.move_v
02.resize_v
03.event_k
04.event_m
05.event_t
06.event_m
07.draw_li
08.put_tex
09.draw_ci
10.draw_el
11.event_draw.py
12.read_image1.py
13.read_image2.py
14.read_image3.py
15.write_image1.py
Position1
Position2
윈도우
import
import
바탕
# 윈도우 이동
cv2.imshow(title1, image)
cv2.imshow(title2, image)
cv2.waitKey(0)
cv2.destroyAllWindows()

❖ WINDOW_AUTOSIZE vs. WINDOW_NORMAL

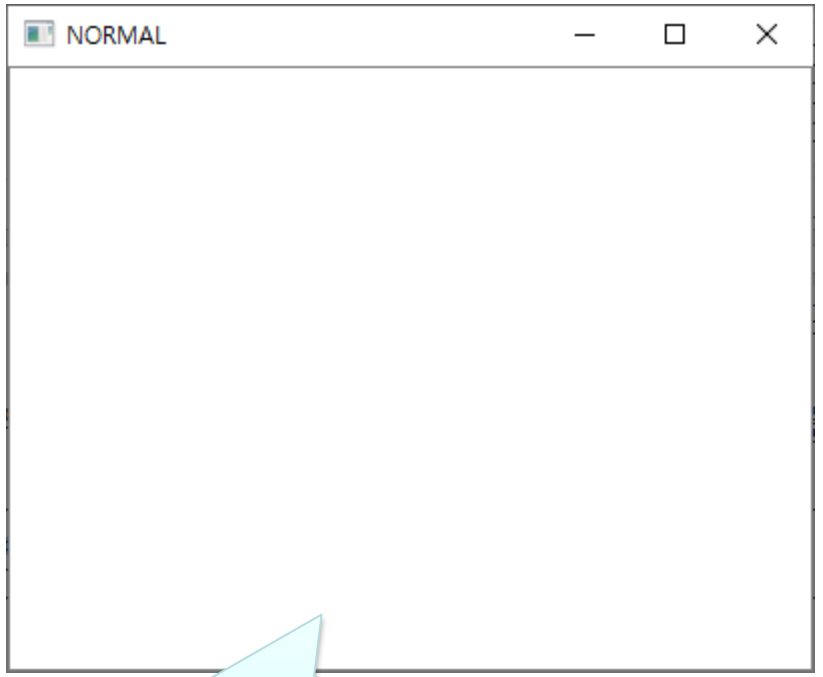
np.ndarray.fill() 함수로 원소값 지정

예제 4.1.2 윈도우의 크기 변경 – 02.window_resize.py

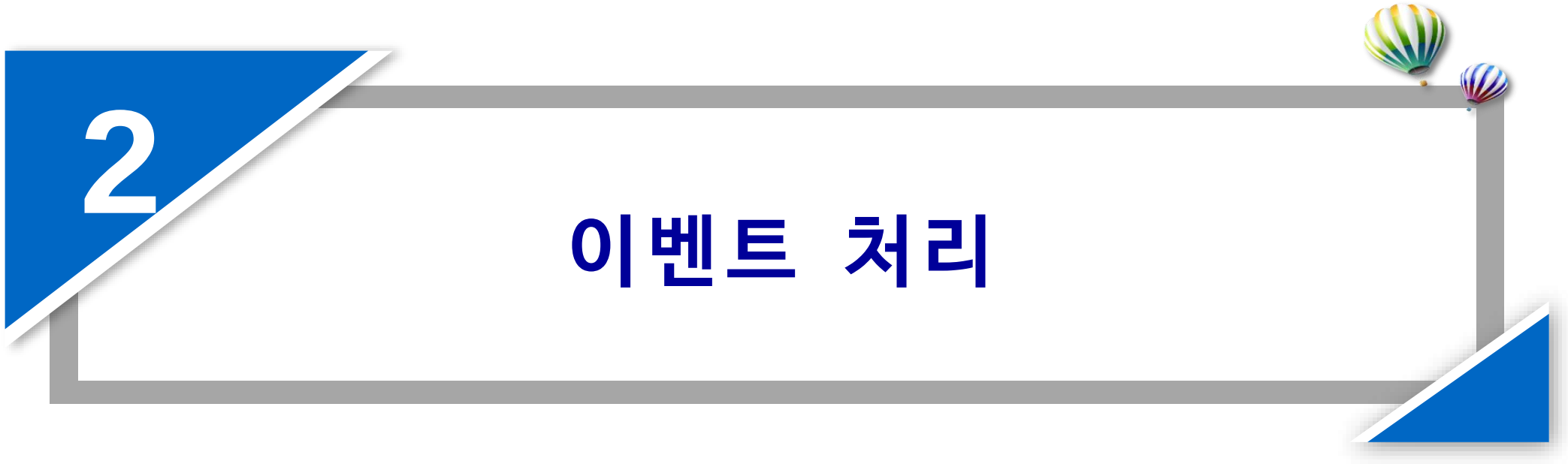
```
01  import numpy as np
02  import cv2
03
04  image = np.zeros((200, 300), np.uint8)              # ndarray 행렬 생성
05  image.fill(255)                                     # 모든 원소에 255(흰색) 지정
06
07  title1, title2 = 'AUTOSIZE', 'NORMAL'               # 윈도우 이름 변수
08  cv2.namedWindow(title1, cv2.WINDOW_AUTOSIZE)        # 윈도우 생성 – 크기변경 불가
09  cv2.namedWindow(title2, cv2.WINDOW_NORMAL)          # 크기 변경 가능
10
11  cv2.imshow(title1, image)                           # 행렬 원소를 영상으로 표시
12  cv2.imshow(title2, image)
13  cv2.resizeWindow(title1, 400, 300)                  # 윈도우 크기 변경
14  cv2.resizeWindow(title2, 400, 300)
15  cv2.waitKey(0)                                      # 키 이벤트(key event) 대기
16  cv2.destroyAllWindows()                             # 열린 모든 윈도우 제거
```



행렬 크기 변경없이 윈도우 크기 변경



행렬 크기 변경되어 보여지며 윈도우 크기 변경, 실제 행렬이 변경되는 것은 아님



2 이벤트 처리

이벤트 처리

❖ 이벤트

- 프로그램에 의해 감지되고 처리될 수 있는 동작이나 사건
- 예
  - 사용자가 키보드의 키를 누르는 것
  - 마우스를 움직인다거나 마우스 버튼을 누르는 것
  - 깊이 들어가면 타이머(timer)와 같은 하드웨어 장치가 발생시키는 이벤트
  - 사용자가 자체적으로 정의하는 이벤트

❖ 일반적으로 이벤트를 처리하기 위해 콜백(callback) 함수 사용

- 콜백 함수
  - 개발자가 시스템 함수를 직접 호출하는 방식
  - 이벤트가 발생하거나 특정 시점에 도달했을 때 시스템이 개발자가 등록한 함수 호출

❖ OpenCV에서도 기본적인 이벤트 처리 함수 지원

- 키보드 이벤트, 마우스 이벤트, 트랙바(trackbar) 이벤트

키보드 이벤트 처리

- **cv.waitKey(delay) → retval**
- **cv.waitKeyEx(delay) → retval**
  - delay(mili-second)만큼 키보드 입력 대기
    - delay <= 0 : 무한대기
    - delay > 0 : 지정된 시간만큼 대기(시간내에 입력 없으면 -1 반환)
  - 이벤트 발생시, 입력된 키 값 반환
    - waitKey() : 1 byte 키 값 반환
    - waiteKeyEx() : 2 byte 키 값 반환
      - 기능키(F1, F2, ..., F12), ↑, ↓, ←, → 화살표 키 등 특수키를 입력 받을 때 사용

Insert	0x2d0000
Delete	0x2e0000
Home	0x240000
End	0x230000
Page Up	0x210000
Page Down	0x220000
←	0x250000
↑	0x260000
→	0x270000
↓	0x280000
F1	0x700000

- **주의 : 키 이벤트를 발생시키려면 윈도우를 반드시 선택(클릭)하여 활성화시킨 후, 키보드의 키를 눌려야 함**

예제 4.2.1 키 이벤트 사용 – 03.event_key.py

```
01  import numpy as np
02  import cv2
03
04  ## switch case문을 사전(dictionary)으로 구현
05  switch_case = {
06        ord('a'): "a키 입력",                    # ord() 함수: 문자 → 아스키코드 변환
07        ord('b'): "b키 입력",
08        0x41: "A키 입력",
09        int('0x42', 16): "B키 입력",             # 0x42(16진수) → 10진수 변환
10        2424832: "왼쪽 화살표키 입력",           # 0x250000
11        2490368: "윗쪽 화살표키 입력",           # 0x260000
12        2555904: "오른쪽 화살표키 입력",         # 0x270000
13        2621440: "아래쪽 화살표키 입력"          # 0x280000
14  }
15
16  image = np.ones((200, 300), np.float)          # 원소값 1인 행렬 생성
17  cv2.namedWindow('Keyboard Event')              # 윈도우 이름
18  cv2.imshow("Keyboard Event", image)
19
20  while True:                                    # 무한 반복
21      key = cv2.waitKeyEx(100)                   # 100ms 동안 키 이벤트 대기
22      if key == 27: break                        # ESC 키 누르면 종료
23
24      try:
25          result = switch_case[key]
26          print(result)
27      except KeyError:
28          result = -1
29
30  cv2.destroyAllWindows()                        # 열린 모든 윈도우 제거
```

마우스 이벤트 처리

❖ cv2.setMouseCallback(winname, onMouse, param=None)

- setMouseCallback() 함수를 사용하여 특정 윈도우를 위한 콜백함수를 지정해 주면 마우스 이벤트가 발생할 때마다 콜백함수(예: onMouse)가 호출됨
  - param : 추가적인 사용자 정의 인수
- 콜백함수의 형식
  - **onMouse(event, x, y, flag, param=None)** # 콜백함수이름이 onMouse인 경우
    - event : 이벤트의 종류
    - x, y : 이벤트가 발생한 마우스 포인터의 좌표
    - param : 마우스 버튼과 동시에 특수키(Shift, Alt, Ctrl)가 눌렸는지 여부 확인

옵션	값	설명
cv2.EVENT_FLAG_LBUTTON	1	왼쪽 버튼 누르기
cv2.EVENT_FLAG_RBUTTON	2	오른쪽 버튼 누르기
cv2.EVENT_FLAG_MBUTTON	4	중간 버튼 누르기
cv2.EVENT_FLAG_CTRLKEY	8	[Ctrl] 키 누르기
cv2.EVENT_FLAG_SHIFTKEY	16	[Shift] 키 누르기
cv2.EVENT_FLAG_ALTKEY	32	[Alt] 키 누르기

옵션	값	설명
cv2.EVENT_MOUSEMOVE	0	마우스 움직임
cv2.EVENT_LBUTTONDOWN	1	왼쪽 버튼 누르기
cv2.EVENT_RBUTTONDOWN	2	오른쪽 버튼 누르기
cv2.EVENT_MBUTTONDOWN	3	중간 버튼 누르기
cv2.EVENT_LBUTTONUP	4	왼쪽 버튼 떼기
cv2.EVENT_RBUTTONUP	5	오른쪽 버튼 떼기
cv2.EVENT_MBUTTONUP	6	중간 버튼 떼기
cv2.EVENT_LBUTTONDBLCLK	7	왼쪽 버튼 더블클릭
cv2.EVENT_RBUTTONDBLCLK	8	오른쪽 버튼 더블클릭
cv2.EVENT_MBUTTONDBLCLK	9	중간 버튼 더블클릭
cv2.EVENT_MOUSEWHEEL	10	마우스 휠
cv2.EVENT_MOUSEHWHEEL	11	마우스 가로 휠

MouseEventType	값	설명
EVENT_MOUSEMOVE	0	마우스가 창 위에서 움직인 경우
EVENT_LBUTTONDOWN	1	마우스 왼쪽 버튼을 누른 경우
EVENT_RBUTTONDOWN	2	마우스 오른쪽 버튼을 누른 경우
EVENT_MBUTTONDOWN	3	마우스 가운데 버튼을 누른 경우
EVENT_LBUTTONUP	4	마우스 왼쪽 버튼을 떼는 경우
EVENT_RBUTTONUP	5	마우스 오른쪽 버튼을 떼는 경우
EVENT_MBUTTONUP	6	마우스 가운데 버튼을 떼는 경우
EVENT_LBUTTONDBLCLK	7	마우스 왼쪽 버튼을 더블클릭하는 경우
EVENT_RBUTTONDBLCLK	8	마우스 오른쪽 버튼을 더블클릭하는 경우
EVENT_MBUTTONDBLCLK	9	마우스 중간 버튼을 더블클릭하는 경우
EVENT_MOUSEWHEEL	10	마우스 휠을 돌리는 경우
EVENT_MOUSEHWHEEL	11	마우스 휠을 좌우로 움직이는 경우

MouseEventType	값	설명
EVENT_FLAG_LBUTTON	1	마우스 왼쪽 버튼이 눌려있음
EVENT_FLAG_RBUTTON	2	마우스 오른쪽 버튼이 눌려있음
EVENT_FLAG_MBUTTON	4	마우스 가운데 버튼이 눌려있음
EVENT_FLAG_CTRLKEY	8	왼쪽 Ctrl 버튼이 눌려있음
EVENT_FLAG_SHIFTKEY	16	Shift 키가 눌려있음
EVENT_FLAG_ALTKEY	32	Alt 키가 눌려있음

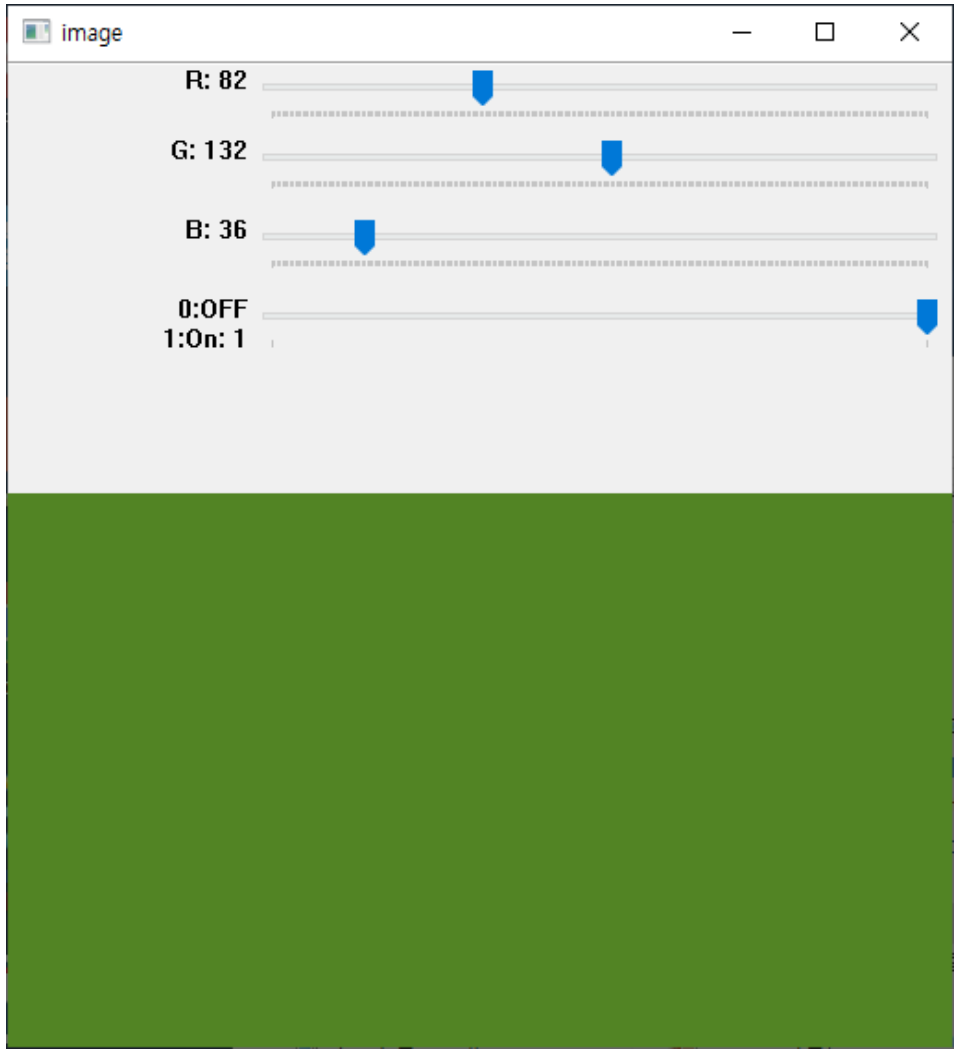
예제 4.2.2 마우스 이벤트 사용 – 04.event_mouse.py

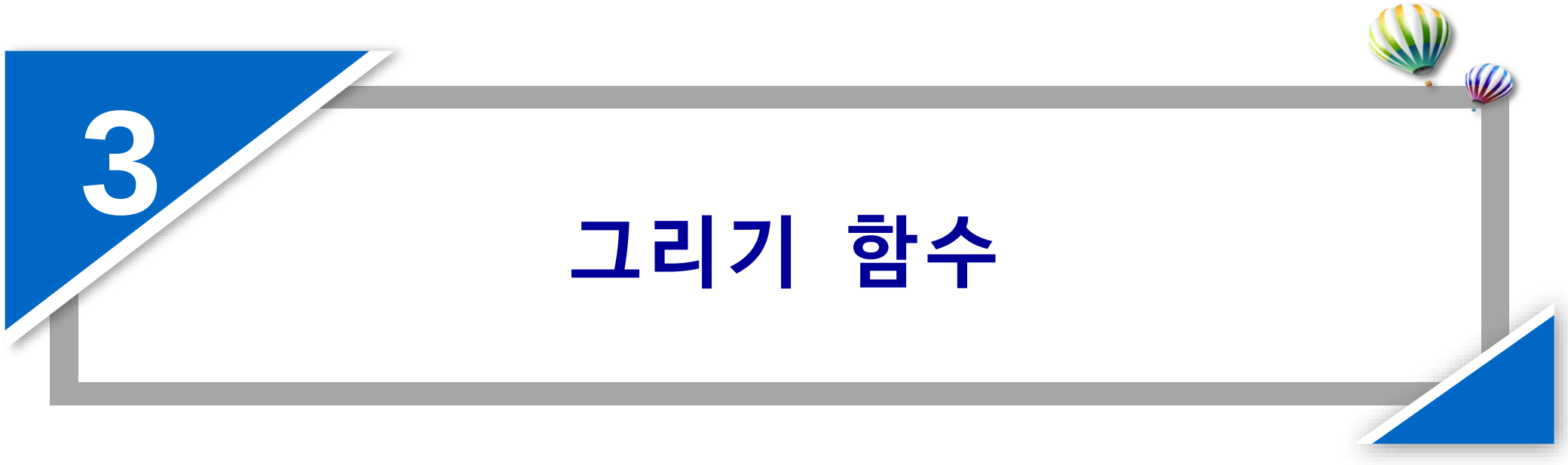
```
01  import numpy as np
02  import cv2
03
04  def onMouse(event, x, y, flags, param):                # 콜백 함수 – 이벤트 내용 출력
05      if event == cv2.EVENT_LBUTTONDOWN:
06          print("마우스 왼쪽 버튼 누르기")
07      elif event == cv2.EVENT_RBUTTONDOWN:
08          print("마우스 오른쪽 버튼 누르기")
09      elif event == cv2.EVENT_RBUTTONUP:
10          print("마우스 오른쪽 버튼 떼기")
11      elif event == cv2.EVENT_LBUTTONDBLCLK:
12          print("마우스 왼쪽 버튼 더블클릭")
13
14  image = np.full((200, 300), 255, np.uint8)             # 초기 영상 생성
15
16  title1, title2 = "Mouse Event1", "Mouse Event2"        # 윈도우 이름
17  cv2.imshow(title1, image)                              # 윈도우 보기
18  cv2.imshow(title2, image)
19
20  cv2.setMouseCallback(title1, onMouse)                  # 마우스 콜백 함수
21  cv2.waitKey(0)                                         # 키 이벤트 대기
22  cv2.destroyAllWindows()                                # 열린 모든 윈도우 제거
```

트랙바 이벤트 제어

- **cv2.createTrackbar(trackbarname, winname, value, count, onChange)**
  - trackbarname : 트랙바 이름
  - winname : 트랙바의 부모 윈도우 이름
  - value : 초기위치값(정수)
  - count : 트랙바의 최대값(최소값은 항상 0)
  - onChange : 콜백함수의 이름(예)
  - **onChange(pos)**
    - pos : 트랙바 슬라이더의 위치
- **cv2.setTrackbarPos(trackbarname, winname, pos)**
- **cv2.getTrackbarPos(trackbarname, winname) → retval**

```python
import cv2
import numpy as np

def nothing(x):
    pass

img = np.zeros((300,512,3), np.uint8)
cv2.namedWindow('image')

# trackbar를 생성하여 named window에 등록
cv2.createTrackbar('R', 'image', 0, 255, nothing)
cv2.createTrackbar('G', 'image', 0, 255, nothing)
cv2.createTrackbar('B', 'image', 0, 255, nothing)
switch = '0:OFF\n1:On'
cv2.createTrackbar(switch, 'image', 1, 1, nothing)

while(1):
    cv2.imshow('image', img)
    if cv2.waitKey(1) & 0xFF == 27:
        break
    r = cv2.getTrackbarPos('R','image')
    g = cv2.getTrackbarPos('G', 'image')
    b = cv2.getTrackbarPos('B', 'image')
    s = cv2.getTrackbarPos(switch, 'image')
    if s == 0:
        img[:] = 0 # 모든 행/열 좌표 값을 0으로 변경. 검은색
    else:
        img[:] = [b,g,r] # 모든 행/열 좌표값을 [b,g,r]로 변경
cv2.destroyAllWindows()
```

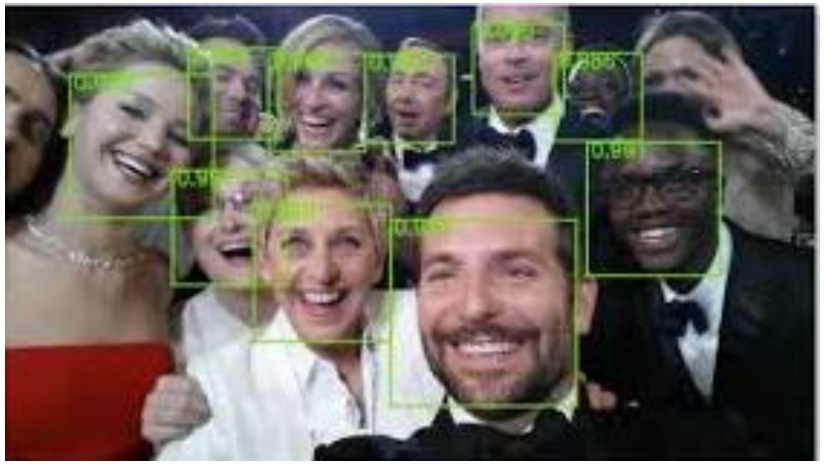



3
그리기 함수

그리기 함수

❖ 영상처리 프로그래밍 과정에서 해당 알고리즘 적용시 결과 확인

- 얼굴 검출 알고리즘을 적용했을 때,
  - 전체 영상 위에 검출한 얼굴 영역을 사각형이나 원으로 표시
- 차선 확인하고자 직선 검출 알고리즘을 적용했을 때,
  - 차선을 정확하게 검출했는지 확인하기 위해 도로 영상 위에 선으로 표시



직선, 사각형 그리기

- **cv2.line(img, pt1, pt2, color[, thickness[, lineType[, shift]]]) → img**

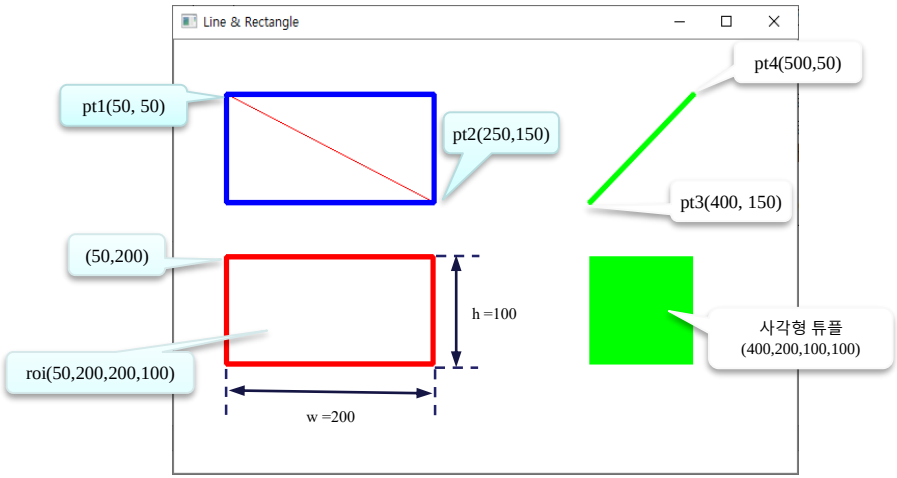
Parameter	내용
img	이미지 파일
pt1	시작점 좌표 (x, y)
pt2	종료점 좌표 (x, y)
color	색상 (blue, green, red) 0 ~ 255
thickness	선 두께 (default 1)
lineType	선 종류 (default cv.Line_8) - LINE_8 : 8-connected line - LINE_4 : 4-connecterd line - LINE_AA : antialiased line
shift	fractional bit (default 0)

❖ cv2.rectangle(img, pt1, pt2, color[, thickness[, lineType[, shift]]]) → img

Parameter	내용
img	이미지 파일
pt1	시작점 좌표 (x, y)
pt2	종료점 좌표 (x, y)
color	색상 (blue, green, red) 0 ~ 255
thickness	선 두께 (default 1)
lineType	선 종류 (default cv.Line_8) - LINE_8 : 8-connected line - LINE_4 : 4-connecterd line - LINE_AA : antialiased line
shift	fractional bit (default 0)

예제 4.3.1 직선 & 사각형 그리기 – 07.draw_line_rect.py

```
01  import numpy as np
02  import cv2
03
04  blue, green, red = (255, 0, 0), (0, 255, 0), (0, 0, 255)          # 색상 선언
05  image = np.zeros((400, 600, 3), np.uint8)                         # 3채널 컬러 영상 생성
06  image[:] = (255, 255, 255)                                        # 3채널 흰색
07
08  pt1, pt2 = (50, 50),   (250, 350)                                 # 좌표 선언 – 정수형 튜플
09  pt3, pt4 = (400, 150), (500, 50)
10  roi = (50, 200, 200, 100)                                         # 사각형 영역 – 4원소 튜플
11
12  ## 직선 그리기
13  cv2.line(image, pt1, pt2, red)
14  cv2.line(image, pt3, pt4, green, 3, cv2.LINE_AA)                  # 계단 현상 감소선
15
16  ## 사각형 그리기
17  cv2.rectangle(image, pt1, pt2, blue, 3, cv2.LINE_4)               # 4방향 연결선
18  cv2.rectangle(image, roi, red, 3, cv2.LINE_8 )                    # 8방향 연결선
19  cv2.rectangle(image, (400, 200, 100, 100), green, cv2.FILLED)     # 내부 채움
20
21  cv2.imshow("Line & Rectangle", image)                             # 윈도우에 영상 표시
22  cv2.waitKey(0)
23  cv2.destroyAllWindows()                                           # 모든 열린 윈도우 닫기
```



텍스트 출력

❖ **cv2.putText(img, text, org, fontFace, fontScale, color[, thickness[, lineType[, bottomLeftOrigin]]]) → None**

parameter	내용
img	이미지 파일
text	출력 문자
org	출력 문자 시작 위치 좌표 (좌측 하단)
fontFace	cv2.FONT_HERSHEY_SIMPLEX : 0 cv2.FONT_HERSHEY_PLAIN : 1 cv2.FONT_HERSHEY_DUPLEX : 2 cv2.FONT_HERSHEY_COMPLEX : 3 cv2.FONT_HERSHEY_TRIPLEX : 4 cv2.FONT_HERSHEY_COMPLEX_SMALL : 5 cv2.FONT_HERSHEY_SCRIPT_SIMPLEX : 6 cv2.FONT_HERSHEY_SCRIPT_COMPLEX : 7 cv2.FONT_ITALIC : 16
fontScale	폰트 크기
color	폰트 색상
thickness	폰트 두께
lineType	선 종류 (default cv.Line_8) - LINE_8 : 8-connected line - LINE_4 : 4-connecterd line - LINE_AA : antialiased line
bottomLeftOrigin	org 사용 옵션. True : 좌측 하단. False : 좌측 상단

표시 문자열

2줄 산세리프 폰트

확대 비율

```
putText(image, "DUPLEX", pt1, FONT_HERSHEY_DUPLEX, 2, Scalar(128, 128, 0));
putText(image, "TRIPLEX", pt2, FONT_HERSHEY_TRIPLEX, 3, Scalar(221, 160, 221));
```

시작좌표(pt1)

시작좌표(pt2)

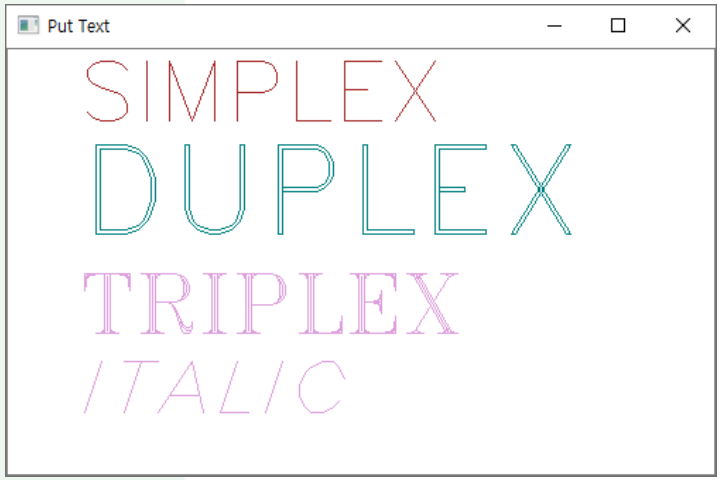
색상

3줄 세리프 폰트

- **세리프 체**
  - 글자의 획 끝에 날카롭게 튀어나온 글자체
  - 명조체, 궁서체 등
- **산세리프 체**
  - 날카로운 장식선이 없는 글자체
  - 돋움체 , 고딕체

예제 4.3.2 글자 쓰기 – 08.put_text.py

```
01  import numpy as np
02  import cv2
03
04  olive, violet, brown = (128, 128, 0), (221, 160, 221), (42, 42, 165)        # 색상 지정
05  pt1, pt2 =  (50, 230), (50, 310)                                            # 문자열 위치 좌표
06
07  image = np.zeros((350, 500, 3), np.uint8)
08  image.fill(255)
09
10  cv2.putText(image, 'SIMPLEX', (50, 50), cv2.FONT_HERSHEY_SIMPLEX, 2, brown)
11  cv2.putText(image, 'DUPLEX', (50, 130), cv2.FONT_HERSHEY_DUPLEX, 3, olive)
12  cv2.putText(image, 'TRIPLEX', pt1, cv2.FONT_HERSHEY_TRIPLEX, 2, violet)
13  fontFace = cv2.FONT_HERSHEY_PLAIN | cv2.FONT_ITALIC     # 글차체 상수
14  cv2.putText(image, 'ITALIC', pt2, fontFace, 4, violet)
15
16  cv2.imshow('Put Text', image)                                               # 윈도우 이름 지정 및 영상 표시
17  cv2.waitKey(0)                                                              # 키이벤트 대기
```



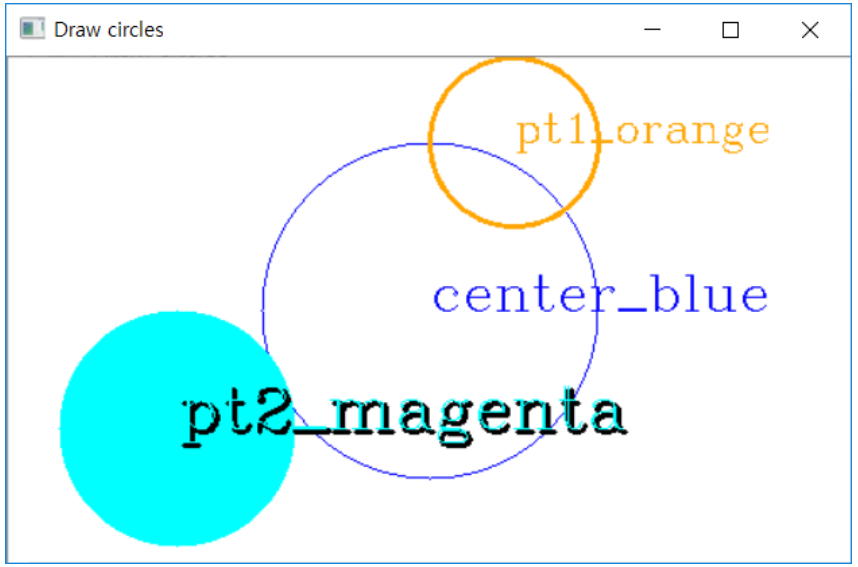
원 그리기

❖ **cv2.circle(img, center, radius, color[, thickness[, lineType[, shift]]]) → img**

Parameter	내용
img	이미지 파일
center	원의 중심 좌표(x, y)
radius	원의 반지름
color	색상 (blue, green, red) 0 ~ 255
thickness	선 두께 (default 1)
lineType	선 종류 (default cv.Line_8) - LINE_8 : 8-connected line - LINE_4 : 4-connecterd line - LINE_AA : antialiased line
shift	fractional bit (default 0)

예제 4.3.3 원 그리기 – 09.draw_circle.py

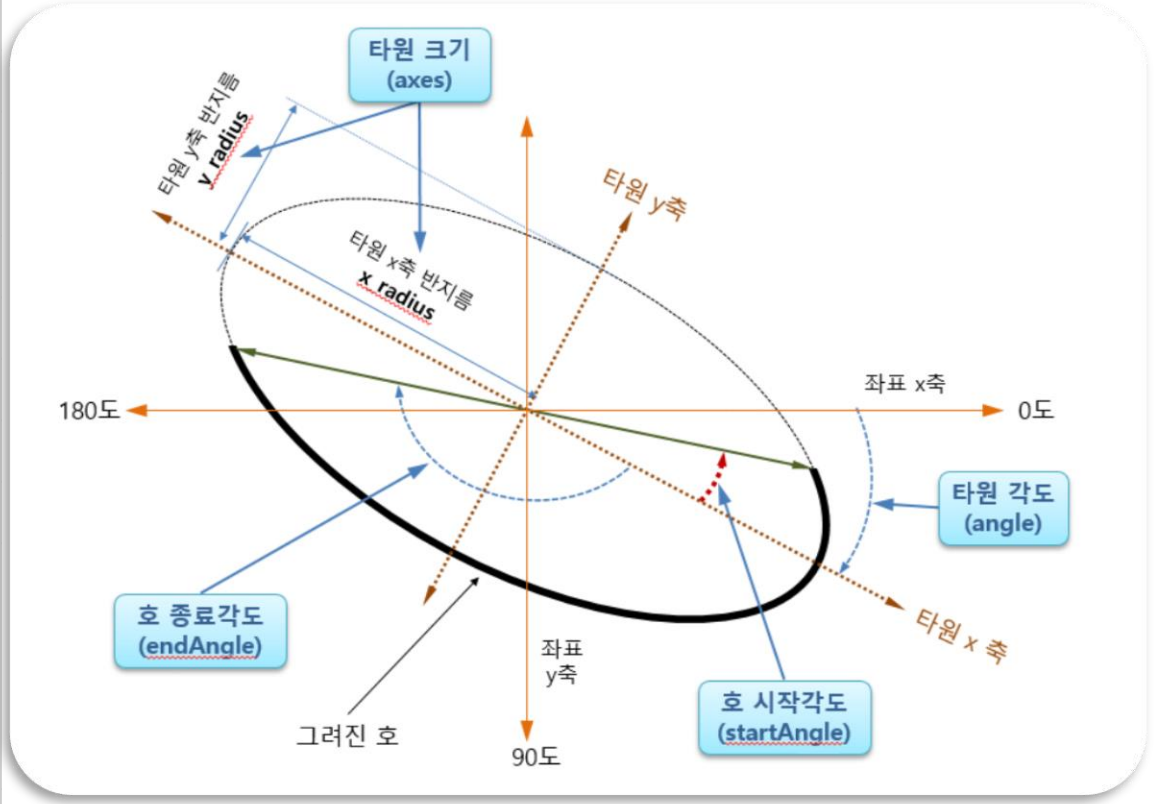
```
01  import numpy as np
02  import cv2
03
04  orange, blue, cyan = (0, 165, 255), (255, 0, 0), (255, 255, 0)
05  white, black = (255, 255, 255), (0, 0, 0)
06  image = np.full((300, 500, 3), white, np.uint8)          # 컬러 영상 생성 및 초기화
07
08  center = (image.shape[1]//2, image.shape[0]//2)          # 영상 중심 좌표 – 역순 구성
09  pt1, pt2 = (300, 50), (100, 220)
10  shade = (pt2[0] + 2, pt2[1] + 2)                         # 그림자 좌표
11
12  cv2.circle(image, center, 100, blue)                     # 원 그리기
13  cv2.circle(image, pt1, 50, orange, 2)
14  cv2.circle(image, pt2, 70, cyan, -1)                     # 원 내부 채움
15
16  font = cv2.FONT_HERSHEY_COMPLEX;
17  cv2.putText(image, 'center_blue', center, font, 1.0, blue)
18  cv2.putText(image, 'pt1_orange', pt1, font, 0.8, orange)
19  cv2.putText(image, 'pt2_cyan', shade, font, 1.2, black, 2)      # 그림자 효과
20  cv2.putText(image, 'pt2_cyan', pt2, font, 1.2, cyan,    1)
21
22  cv2.imshow("Draw circles", image)
23  cv2.waitKey(0)
```



타원 그리기

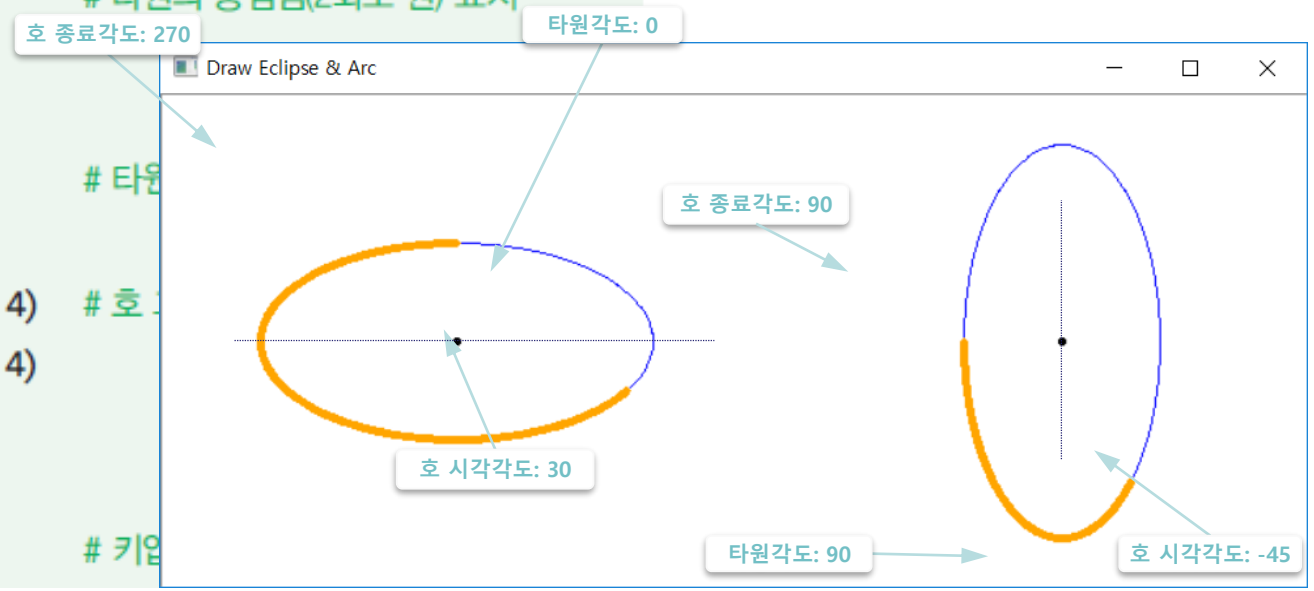
❖ **cv2.ellipse(img, center, axes, angle, startAngle, endAngle, color[, thickness[, lineType[, shift]]]) → img**

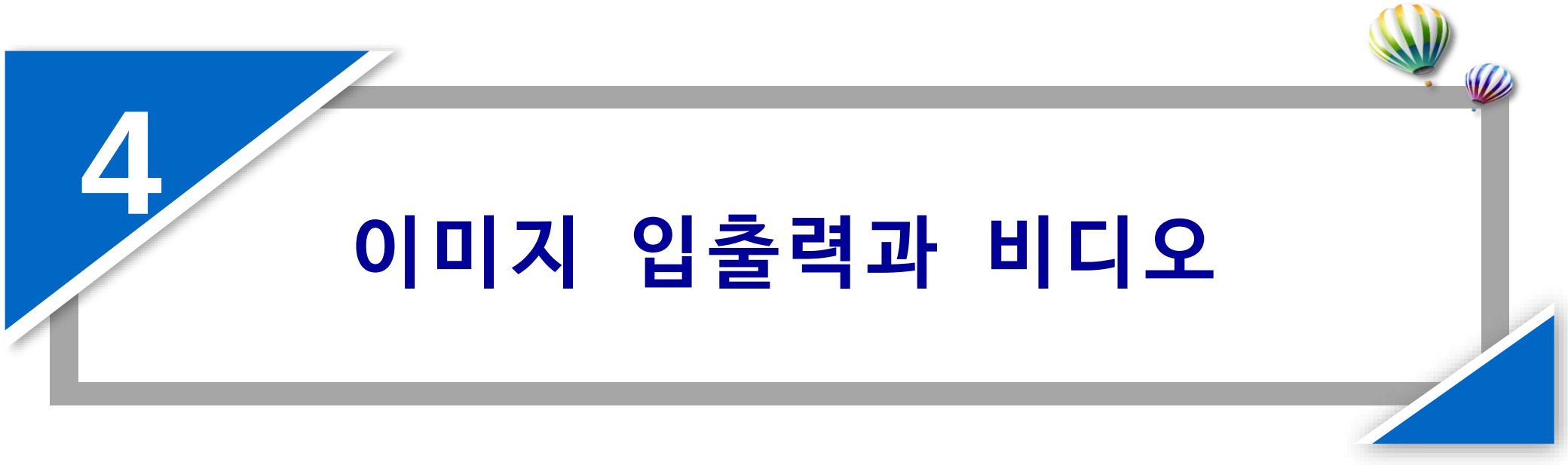
parameter	내용
img	이미지 파일
center	타원의 중심 좌표(x, y)
axes	축의 절반 길이(중심에서 긴 거리, 짧은 거리)
angle	타원의 기울기
startAngle	타원을 그리는 시작 각도
endAngle	타원을 그리는 종료 각도
color	색상 (blue, green, red) 0 ~ 255
thickness	선 두께 (default 1)
lineType	선 종류 (default cv.Line_8) - LINE_8 : 8-connected line - LINE_4 : 4-connecterd line - LINE_AA : antialiased line
shift	fractional bit (default 0)



예제 4.3.4 타원 및 호 그리기 – 10.draw_ellipse.py

```
01  import numpy as np
02  import cv2
03
04  orange, blue, white = (0, 165, 255), (255, 0, 0), (255,255,255) # 색상 지정
05  image = np.full((300, 700, 3), white, np.uint8)          # 3채널 행렬 생성 및 초기화
06
07  pt1, pt2 = (180, 150), (550, 150)                        # 타원 중심점
08  size = (120, 60)                                         # 타원 크기 – 반지름 값임
09
10  cv2.circle(image, pt1, 1, 0, 2)                          # 타원의 중심점(2화소 원) 표시
11  cv2.circle(image, pt2, 1, 0, 2)
12
13  cv2.ellipse(image, pt1, size,  0, 0, 360, blue, 1)       # 타원
14  cv2.ellipse(image, pt2, size, 90, 0, 360, blue, 1)
15  cv2.ellipse(image, pt1, size,  0, 30, 270, orange, 4)    # 호
16  cv2.ellipse(image, pt2, size, 90, -45, 90, orange, 4)
17
18  cv2.imshow("문자열", image)
19  cv2.waitKey()                                            # 키입
```





4 이미지 입출력과 비디오

이미지 읽기와 쓰기

❖ cv2.imread(filename[, flags]) → retval

parameter	내용
filename	읽어올 파일 명
flags	cv2.IMREAD_UNCHAGED : alpha channel 까지 포함해 읽음(-1) cv2.IMREAD_GRAYSCALE : Grayscale로 읽음(0) cv2.IMREAD_COLOR : Color로 읽음(1) cv2.IMREAD_ANYDEPTH : 정의된 depth에 따라 16/32비트로 변환 (기본 8비트)(2) cv2.IMREAD_ANYCOLOR : 입력파일에 정의된 타입으로 변환(4)

❖ cv2.imwrite(filename, img[, params]) → retval

parameter	내용
filename	저장할 파일 명
img	이미지 파일
params	cv2.IMWRITE_JPEG_QUALITY : JPEG 포맷으로 저장 cv2.IMWRITE_PNG_COMPRESSION : PNG 압축레벨로 저장 cv2.IMWRITE_PXM_BINARY : PPM, PGM 파일의 이진 포맷

예제 4.4.3 행렬 영상 저장1 – 15.write_image1.py

```
01  import cv2
02
03  image = cv2.imread("images/read_color.jpg", cv2.IMREAD_COLOR)
04  if image is None: raise Exception("영상파일 읽기 에러")          # 예외처리
05
06  params_jpg = (cv2.IMWRITE_JPEG_QUALITY, 10)                    # JPEG 화질 설정
07  params_png = [cv2.IMWRITE_PNG_COMPRESSION, 9]                  # PNG 압축 레벨 설정
08
09  ## 행렬을 영상파일로 저장
10  cv2.imwrite("images/write_test1.jpg", image)                   # 디폴트는 95
11  cv2.imwrite("images/write_test2.jpg", image, params_jpg)       # 지정한 화질로 저장
12  cv2.imwrite("images/write_test3.png", image, params_png)
13  cv2.imwrite("iamges/write_test4.bmp", image)                   # BMP 파일로 저장
14  print("저장 완료")
```

심화예제 4.4.4 행렬 영상 저장2 – 16.write_image2.py

```
01  import numpy as np
02  import cv2
03
04  image8 = cv2.imread("images/read_color.jpg", cv2.IMREAD_COLOR)
05  if image8 is None: raise Exception("영상파일 읽기 에러")          # 영상파일 예외처리
06
07  image16 = np.uint16(image8 * (65535/255))       # 형변환 및 화소 스케일 조정
08  image32 = np.float32(image8 * (1/255))
09
10  ## 화소값 확인 – 관심 영역((10, 10) 위치에서 2행, 3열) 출력
11  print("image8 행렬의 일부\n %s\n"  % image8[10:12, 10:13])
12  print("image16 행렬의 일부\n %s\n" % image16[10:12, 10:13])
13  print("image32 행렬의 일부\n %s\n" % image32[10:12, 10:13])
14
15  cv2.imwrite('images/write_test_16.tif', image16)          # 16비트 행렬 저장
16  cv2.imwrite('images/write_test_32.tif', image32)          # 32비트 행렬 저정
17
18  cv2.imshow('image16', image16)          # 영상 표시
19  cv2.imshow('image32', (image32*255).astype('uint8'))
20  cv2.waitKey(0)
```

비디오 읽기

- **cv2.VideoCapture(filename) → <VideoCapture object>**
- **cv2.VideoCapture(device) → <VideoCapture object>**
  - cv2.VideoCapture.**open**(filename) → retval
  - cv2.VideoCapture.**open**(device) → retval
  - cv2.VideoCapture.**isOpened**() → retval
  - cv2.VideoCapture.**get**(propId) → retval
  - cv2.VideoCapture.**set**(propId, value) → retval
  - cv2.VideoCapture.**read**([image]) → retval, image
  - cv2.VideoCapture.**retrieve**([image[, flag]]) → retval, image
  - cv2.VideoCapture.**grab**() → retval
  - cv2.VideoCapture.**release**() → None

❖ 속성 – get(), set()

parameter	내용
cv2.CAP_PROP_POS_MSEC	동영상 파일의 현재위치(milli-second)
cv2.CAP_PROP_POS_FRAMES	캡쳐되는 프레임의 번호
cv2.CAP_PROP_POS_AVI_RATIO	동영상 파일의 상대적 위치(0: 시작, 1: 끝)
cv2.**CAP_PROP_FRAME_WIDTH**	프레임의 너비
cv2.**CAP_PROP_FRAME_HEIGHT**	프레임의 높이
cv2.**CAP_PROP_FPS**	초당 프레임 수
cv2.**CAP_PROP_FOURCC**	코덱을 나타내는 4개의 문자
cv2.**CAP_PROP_FRAME_COUNT**	총 프레임 수
cv2.CAP_PROP_FORMAT	retrieve()에 의해 반환되는 행렬의 포맷
cv2.CAP_PROP_BRIGHTNESS	Brightness of the image (only for cameras)
cv2.CAP_PROP_CONTRAST	Contrast of the image (only for cameras)
cv2.CAP_PROP_SATURATION	Saturation of the image (only for cameras)
cv2.CAP_PROP_HUE	Hue of the image (only for cameras)
cv2.CAP_PROP_GAIN	Gain of the image (only for cameras)
cv2.CAP_PROP_EXPOSURE	Exposure(노출) (only for cameras)
cv2.CAP_PROP_CONVERT_RGB	Boolean flags indicating whether images should be converted to RGB

비디오 쓰기

❖ cv2.VideoWriter([filename, fourcc, fps, frameSize[, isColor]]) → <VideoWriter object>

parameter	내용
filename	저장할 동영상 파일명
fourcc	frame 압축 관련 4자리 code
fps	초당 저장할 frame
frameSize	frame size (가로, 세로)
isColor	컬러 저장 여부

- cv2.VideoWriter.**open**(filename, fourcc, fps, frameSize[, isColor]) → retval
- cv2.VideoWriter.**isOpened**() → retval
- cv2.VideoWriter.**write**(image) → None
- cv2.**VideoWriter_fourcc**(c1, c2, c3, c4) → retval
  - (예)
    - fourcc = cv2.VideoWriter_fourcc(*'DIVX')
    - out = cv2.VideoWriter(videoFile1, fourcc, 25.0, (320,240)

〈표 4.5.2〉 주요 코덱 문자

속성 상수	설명
cv2.VideoWriter_fourcc('D', 'I', 'V', '4') cv2.VideoWrite_fourcc(*'DIV4')	DivX MPEG-4
cv2.VideoWriter_fourcc('D', 'I', 'V', '5') cv2.VideoWrite_fourcc(*'DIV5')	Div5
cv2.VideoWriter_fourcc('D', 'I', 'V', 'X') cv2.VideoWrite_fourcc(*'DIVX')	DivX
cv2.VideoWriter_fourcc('D', 'X', '5', '0') cv2.VideoWrite_fourcc(*'DX50')	DivX MPEG-4
cv2.VideoWriter_fourcc('F', 'M', 'P', '4') cv2.VideoWrite_fourcc(*'FMP4')	FFMpeg
cv2.VideoWriter_fourcc('I', 'Y', 'U', 'V') cv2.VideoWrite_fourcc(*'TYUV')	IYUV
cv2.VideoWriter_fourcc('M', 'J', 'P', 'G') cv2.VideoWrite_fourcc(*'MJPG')	Motion JPEG codec
cv2.VideoWriter_fourcc'M', 'P', '4', '2') cv2.VideoWrite_fourcc(*'MP42')	MPEG4 v2
cv2.VideoWriter_fourcc('M', 'P', 'E', 'G') cv2.VideoWrite_fourcc(*'MPEG')	MPEG codecs
cv2.VideoWriter_fourcc('X', 'V', 'I', 'D') cv2.VideoWrite_fourcc(*'XVID')	XVID codecs
cv2.VideoWriter_fourcc('X', '2', '6', '4') cv2.VideoWrite_fourcc(*'X264')	H.264/AVC codecs
-1	코덱 선택 대화상자 띄움

예제 4.5.1 카메라 프레임 읽기 – 17.read_pccamera.py

```
01  import cv2
02
03  def put_string(frame, text, pt, value, color=(120, 200, 90) ):   # 문자열 출력 함수
04      text += str(value)
05      shade = (pt[0] + 2, pt[1] + 2)
06      font = cv2.FONT_HERSHEY_SIMPLEX
07      cv2.putText(frame, text, shade, font, 0.7, (0, 0, 0), 2)      # 그림자 효과
08      cv2.putText(frame, text, pt    , font, 0.7, color, 2)         # 글자 적기
09
10  capture = cv2.VideoCapture(0)                                     # 0번 카메라 연결
11  if capture.isOpened() == False:                                   # 카메라 연결 예외처리
12       raise Exception("카메라 연결 안됨")
13
14  ## 카메라 속성 획득 및 출력
15  print("너비 %d" % capture.get(cv2.CAP_PROP_FRAME_WIDTH))
16  print("높이 %d" % capture.get(cv2.CAP_PROP_FRAME_HEIGHT))
17  print("노출 %d" % capture.get(cv2.CAP_PROP_EXPOSURE))
18  print("밝기 %d" % capture.get(cv2.CAP_PROP_BRIGHTNESS))
19
```

```
Run: 17.read_pccamera
C:\Python\python.exe D:/source/chap04/17.read_pccamera.py
Traceback (most recent call last):
  File "D:/source/chap04/17.read_pccamera.py", line 11, in <module>
    if capture.isOpened() == False: raise Exception("카메라 연결 안됨")
Exception: 카메라 연결 안됨
```

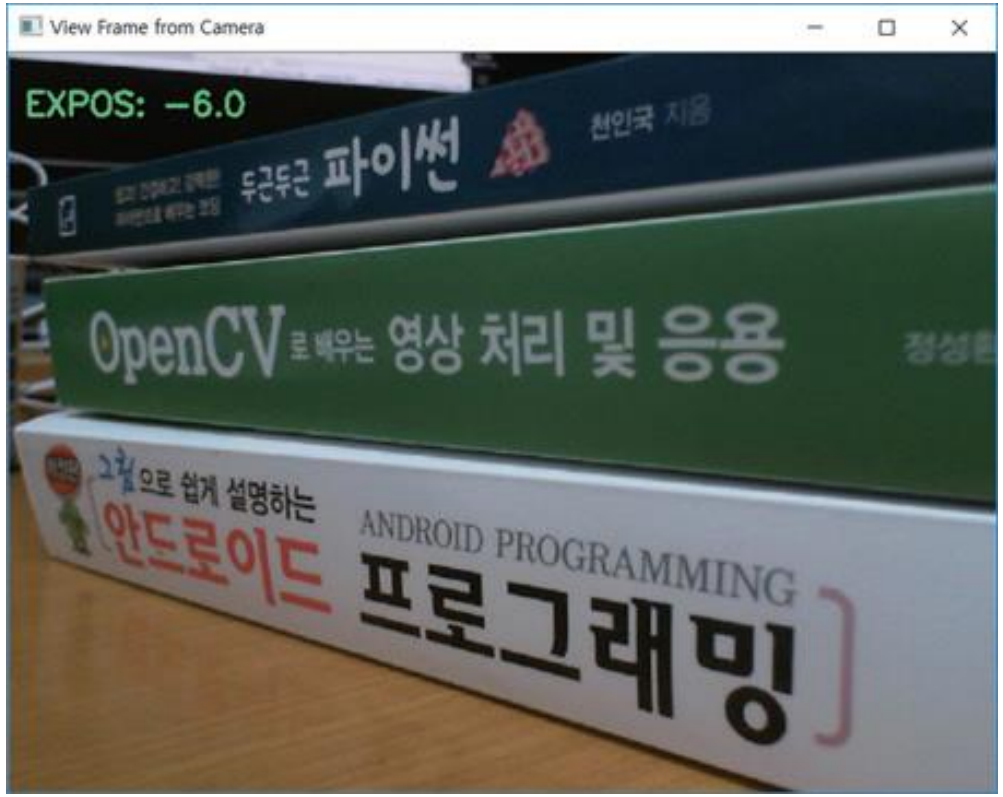
continue…

```
14  ## 카메라 속성 획득 및 출력
15  print("너비 %d" % capture.get(cv2.CAP_PROP_FRAME_WIDTH))
16  print("높이 %d" % capture.get(cv2.CAP_PROP_FRAME_HEIGHT))
17  print("노출 %d" % capture.get(cv2.CAP_PROP_EXPOSURE))
18  print("밝기 %d" % capture.get(cv2.CAP_PROP_BRIGHTNESS))
19
20  while True                                          # 무한 반복
21      ret, frame = capture.read()                     # 카메라 영상 받기
22      if not ret: break
23      if cv2.waitKey(30) >= 0: break                  # 종료 조건 - 스페이스바 키
24
25      exposure = capture.get(cv2.CAP_PROP_EXPOSURE)   # 노출 속성 획득
26      put_string(frame, 'EXPOS: ', (10, 40), exposure)
27      title = "View Frame from Camera"
28      cv2.imshow(title, frame)                        # 윈도우에 영상 띄우기
29  capture.release()
```

카메라에서 프레임 읽기

❖ 실행결과

```
Run:  17.read_pccamera
C:\Python\python.exe D:/source/chap04/17.read_pccamera.py
너비 640
높이 480
노출 -6
밝기 143
```



continue…

심화예제 4.5.2 카메라 속성 설정 – 18.set_camera_attr.py

저자 생성 모듈의 함수 임포트

```
01  import cv2
02  from Common.utils import put_string                    # 함수 재사용 위한 임포트
03
04  def zoom_bar(value):                                   # 줌 조절 콜백 함수
05      global capture
06      capture.set(cv2.CAP_PROP_ZOOM, value)              # 줌 설정
07
08  def focus_bar(value):                                  # 초점조절 콜백 함수
09      global capture
10      capture.set(cv2.CAP_PROP_FOCUS, value)
11
12  capture = cv2.VideoCapture(0)                          # 0번 카메라 연결
13  if capture.isOpened() == False: raise Exception("카메라 연결 안됨")      # 예외처리
14
15  capture.set(cv2.CAP_PROP_FRAME_WIDTH, 400)             # 카메라 프레임 너비
16  capture.set(cv2.CAP_PROP_FRAME_HEIGHT, 300)            # 카메라 프레임 높이
17  capture.set(cv2.CAP_PROP_AUTOFOCUS, 0)                 # 자동초점 중지
18  capture.set(cv2.CAP_PROP_BRIGHTNESS, 100)              # 프레임 밝기 초기화
19
20  title = "Change Camera Properties"                     # 윈도우 이름 지정
21  cv2.namedWindow(title)                                 # 윈도우 생성
22  cv2.createTrackbar('zoom', title, 0, 10, zoom_bar)     # 줌 트랙바
23  cv2.createTrackbar('focus', title, 0, 40, focus_bar)   # 포커스 트랙바
```

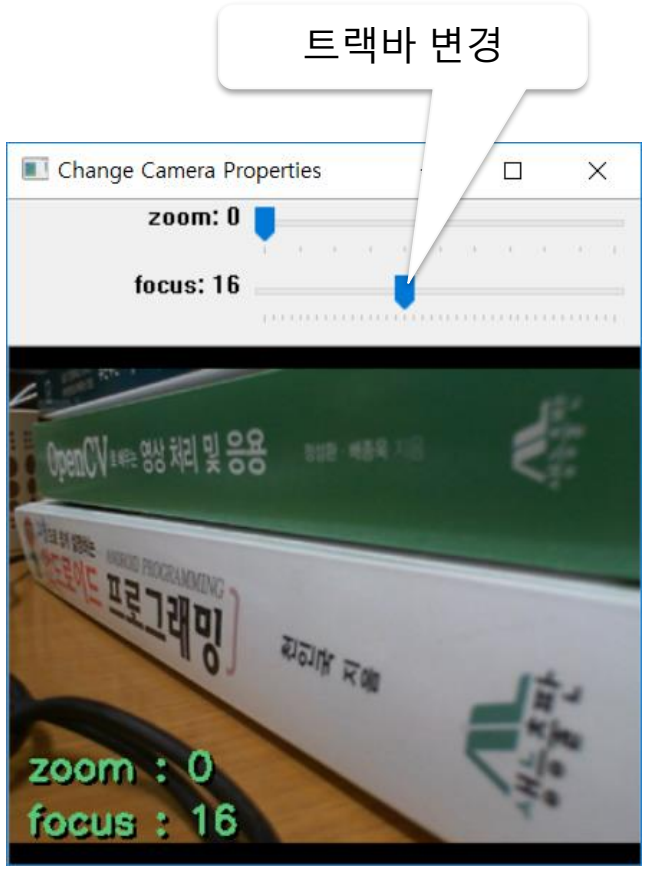
전역변수

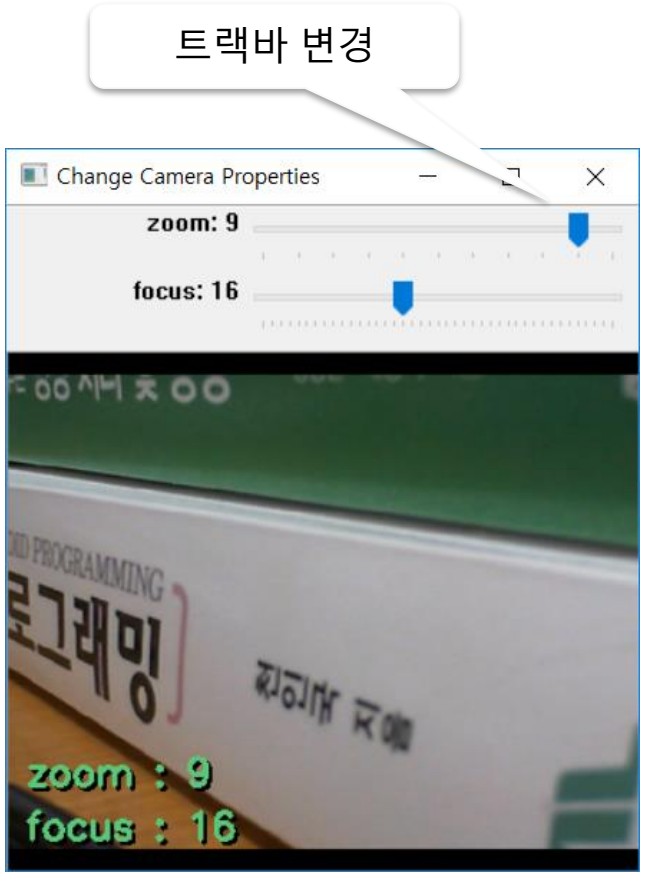
continue…

```
25  while True:
26      ret, frame = capture.read()                    # 카메라 영상 받기
27      if not ret: break
28      if cv2.waitKey(30) >= 0: break
29
30      zoom = int(capture.get(cv2.CAP_PROP_ZOOM))     # 카메라 속성 가져오기
31      focus = int(capture.get(cv2.CAP_PROP_FOCUS))
32      put_string(frame, 'zoom : ', (10, 240), zoom)  # 줌 값 표시
33      put_string(frame, 'focus : ', (10, 270), focus) # 초점 값 표시
34      cv2.imshow(title, frame)
35
36  capture.release()                                  # 비디오 캡쳐 메모리 해제
```

❖ 실행결과


Change Camera Properties
zoom: 0
focus: 29
zoom : 0
focus : 29


트랙바 변경
Change Camera Properties
zoom: 0
focus: 16
zoom : 0
focus : 16


트랙바 변경
Change Camera Properties
zoom: 9
focus: 16
zoom : 9
focus : 16

카메라 프레임을 동영상파일로 저장

❖ 카메라 영상 실시간 저장 방법

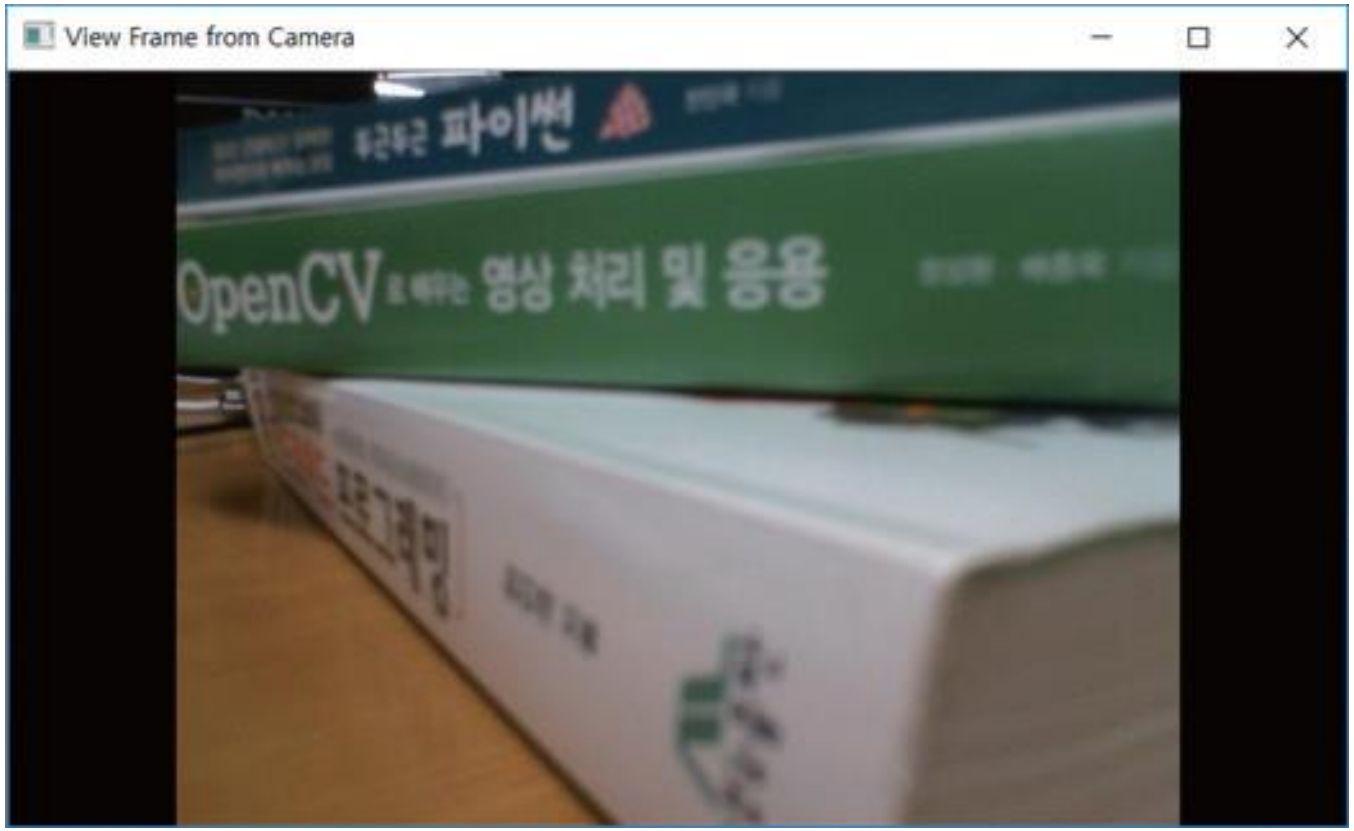
예제 4.5.3 카메라 프레임을 동영상파일로 저장 – 19.write_camera_frame.py

```
01  import cv2
02
03  capture = cv2.VideoCapture(0)                          # 0번 카메라 연결
04  if capture.isOpened() == False: raise Exception("카메라 연결 안됨")
05
06  fps = 29.97                                            # 초당 프레임 수
07  delay = round(1000/fps)                                # 프레임 간 지연 시간
08  size = (640, 360)                                      # 동영상파일 해상도
09  fourcc = cv2.VideoWriter_fourcc(*'DX50')               # 압축 코덱 설정
10
11  ## 카메라 속성 실행창에 출력
12  print("width × height: ", size )
13  print("VideoWriterfourcc: %s" % fourcc)
14  print("delay: %2d ms" % delay)
15  print("fps: %.2f" % fps)
16
17  capture.set(cv2.CAP_PROP_ZOOM, 1)                      # 카메라 속성 지정
```

```
18  capture.set(cv2.CAP_PROP_FOCUS, 0)
19  capture.set(cv2.CAP_PROP_FRAME_WIDTH, size[0]) # 해상도 설정
20  capture.set(cv2.CAP_PROP_FRAME_HEIGHT, size[1])
21
22  ## 동영상파일 개방 및 코덱, 해상도 설정
23  writer = cv2.VideoWriter("images/video_file.avi", fourcc, fps, size)
24  if writer.isOpened() == False: raise Exception("동영상파일 개방 안됨")
25
26  while True:
27      ret, frame = capture.read()              # 카메라 영상 받기
28      if not ret: break
29      if cv2.waitKey(delay) >= 0: break
30
31      writer.write(frame)                      # 프레임을 동영상으로 저장
32      cv2.imshow("View Frame from Camera", frame)
33
34  writer.release()
35  capture.release()
```

❖ 실행결과

```
Run:  19.write_camera_frame
C:\Python\python.exe D:/source/chap04/19.write_camera_frame.py
프레임 해상도: (640, 360)
압축코덱 숫자: 808802372
delay: 33 ms
fps: 29.97
```



동영상파일 읽기

〈표 4.5.3〉 프레임 번호에 따른 영상처리 예시

프레임 번호	영상처리
1 ~ 99	아무런 영상처리를 적용하지 않음
100 ~ 199	프레임별 화소의 파란색 성분에 100을 더해서 영상을 더 푸르게 만듦
200 ~ 299	프레임별 화소의 녹색 성분에 100을 더해서 영상을 더 녹색으로 만듦
300 ~ 399	프레임별 화소의 빨간색 성분에 100을 더해서 영상을 더 빨갛게 만듦

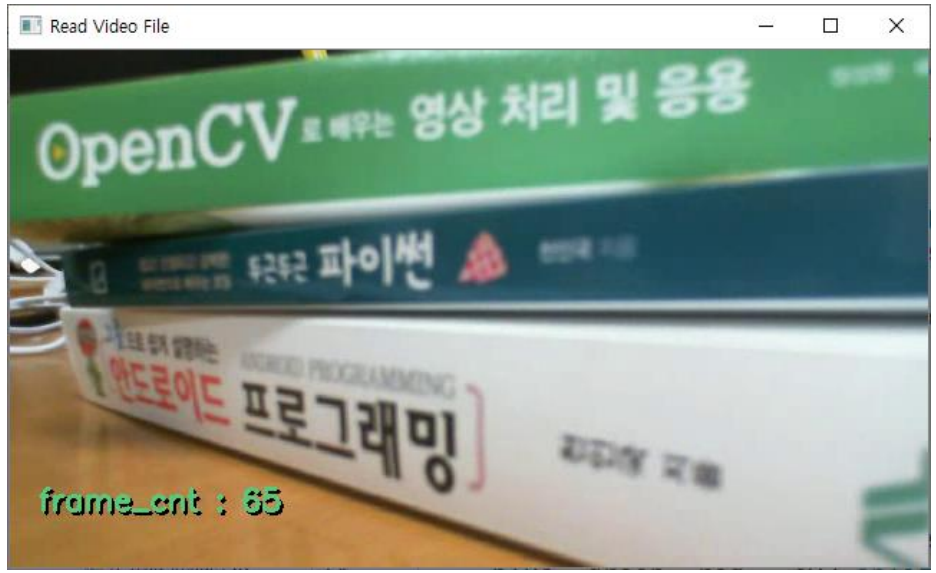
예제 4.5.4 동영상파일 읽기 – 20.read_video_file.py

```
01  import cv2
02  from Common.utils import put_string                    # 글쓰기 함수 임포트
03
04  capture = cv2.VideoCapture("images/video_file.avi")    # 동영상파일 개방
05  if not capture.isOpened(): raise Exception("동영상파일 개방 안됨")          # 예외 처리
06
07  frame_rate = capture.get(cv2.CAP_PROP_FPS)             # 초당 프레임 수
08  delay = int(1000 / frame_rate)                         # 지연 시간
09  frame_cnt = 0                                          # 현재 프레임 번호
```

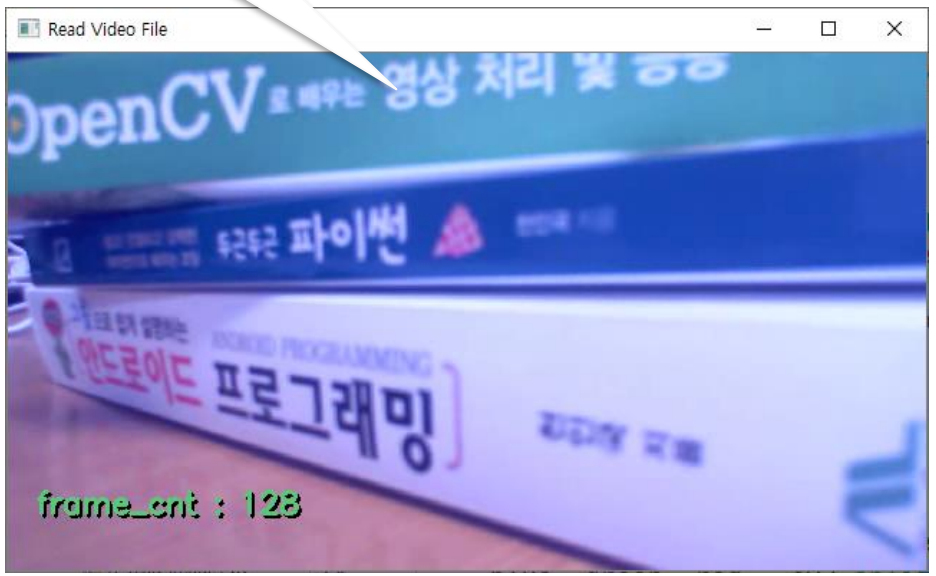
```
11  while True:
12      ret, frame = capture.read()
13      if not ret or cv2.waitKey(delay) >= 0: break      # 프레임 간 지연 시간 지정
14      blue, green, red = cv2.split(frame)               # 컬러 영상 채널 분리
15      frame_cnt += 1
16
17      if 100 <= frame_cnt < 200: cv2.add(blue, 100, blue)        # blue 채널 밝기 증가
18      elif 200 <= frame_cnt < 300: cv2.add(green, 100, green)    # green 채널 밝기 증가
19      elif 300 <= frame_cnt < 400: cv2.add(red  , 100, red)      # red 채널 밝기 증가
20
21      frame = cv2.merge( [blue, green, red] )                    # 단일채널 영상 합성
22      put_string(frame, 'frame_cnt: ', (20, 30), frame_cnt)
23      cv2.imshow("Read Video File", frame)
24  capture.release()
```

continue…

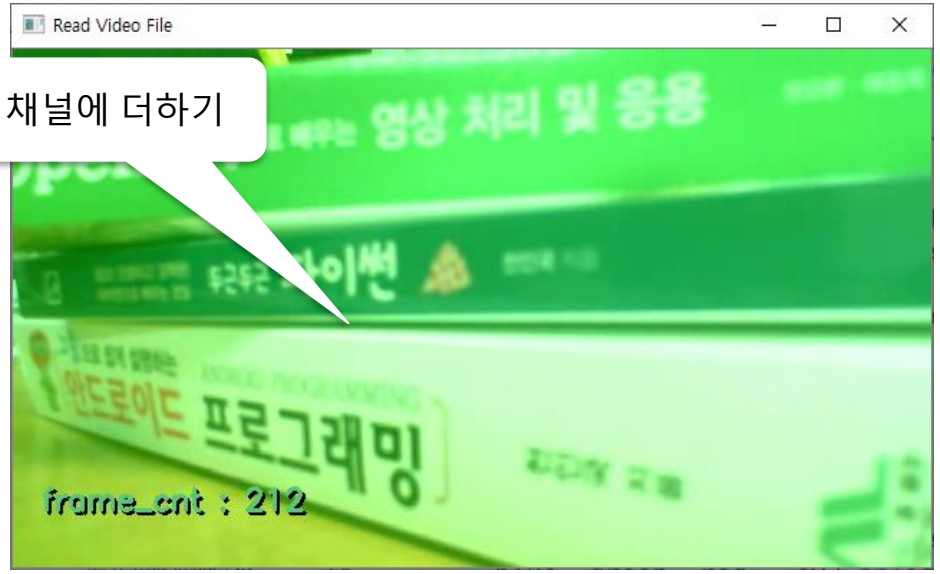
❖ 실행결과



blue 채널에 더하기



G 채널에 더하기



R 채널에 더하기